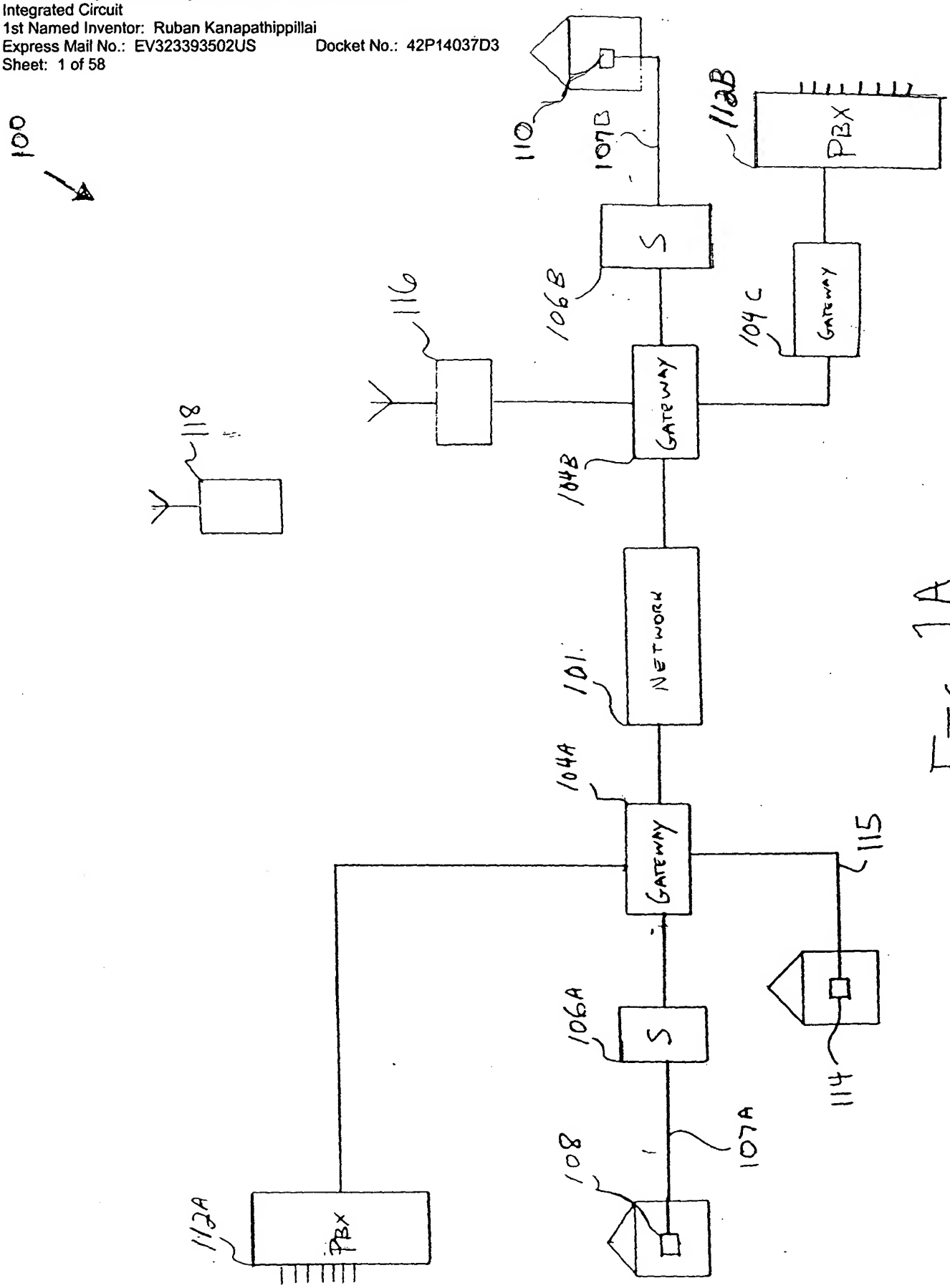

100
110
107B
112B
PBX
S
106B
GATEWAY
104C
116
118
GATEWAY
104B
NETWORK
101
104A
GATEWAY
115
S
106A
114
108
107A
112A
PBX

FIG. 1A

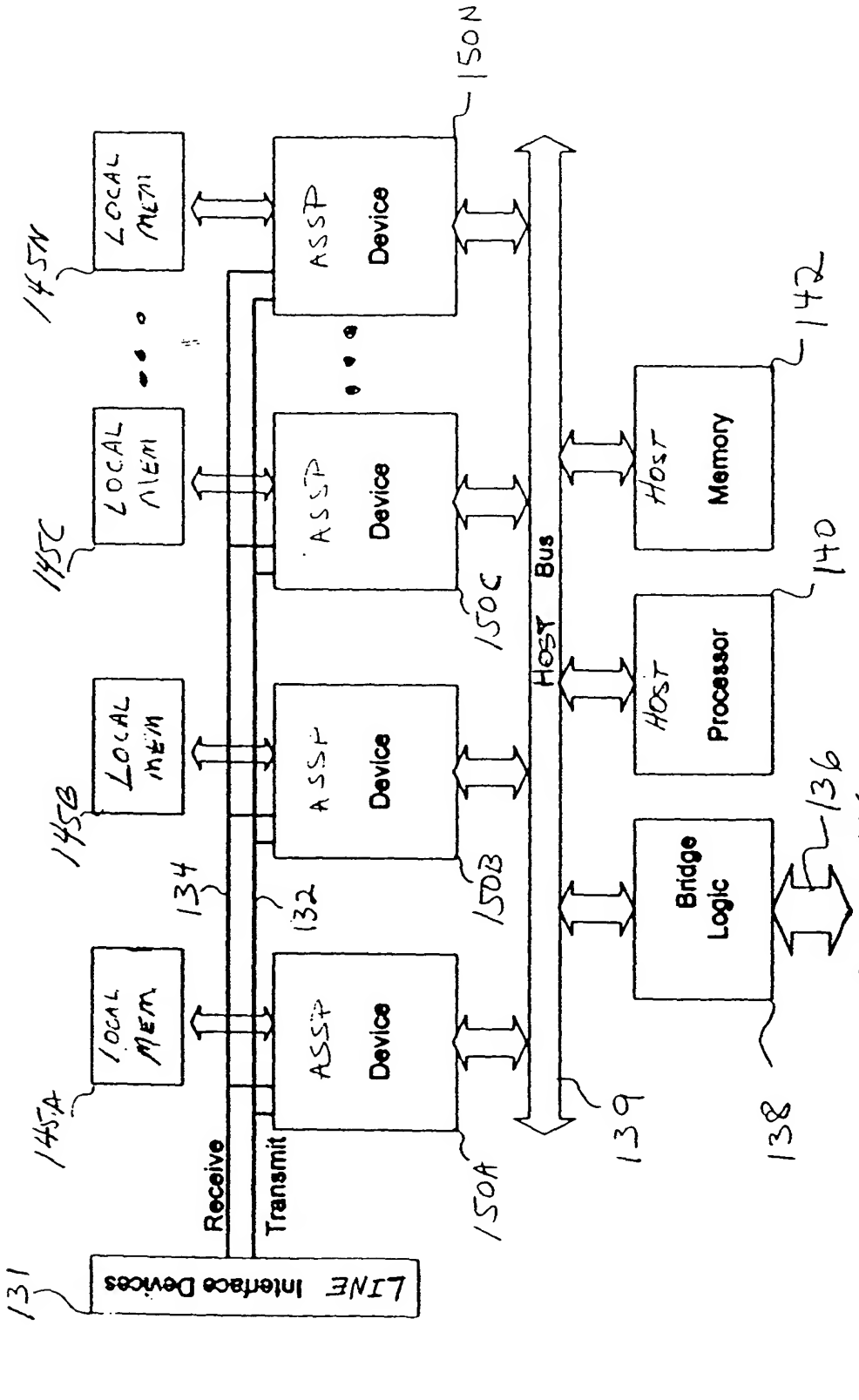

LINE Interface Devices
131
Receive
Transmit
134
132
Local Mem
145A
145B
145C
145N
ASSP Device
150A
150B
150C
150N
Host Bus
139
Bridge Logic
138
Host Processor
140
Host Memory
142
136
Backplane Network Interface Bus
130

FIG. 1B

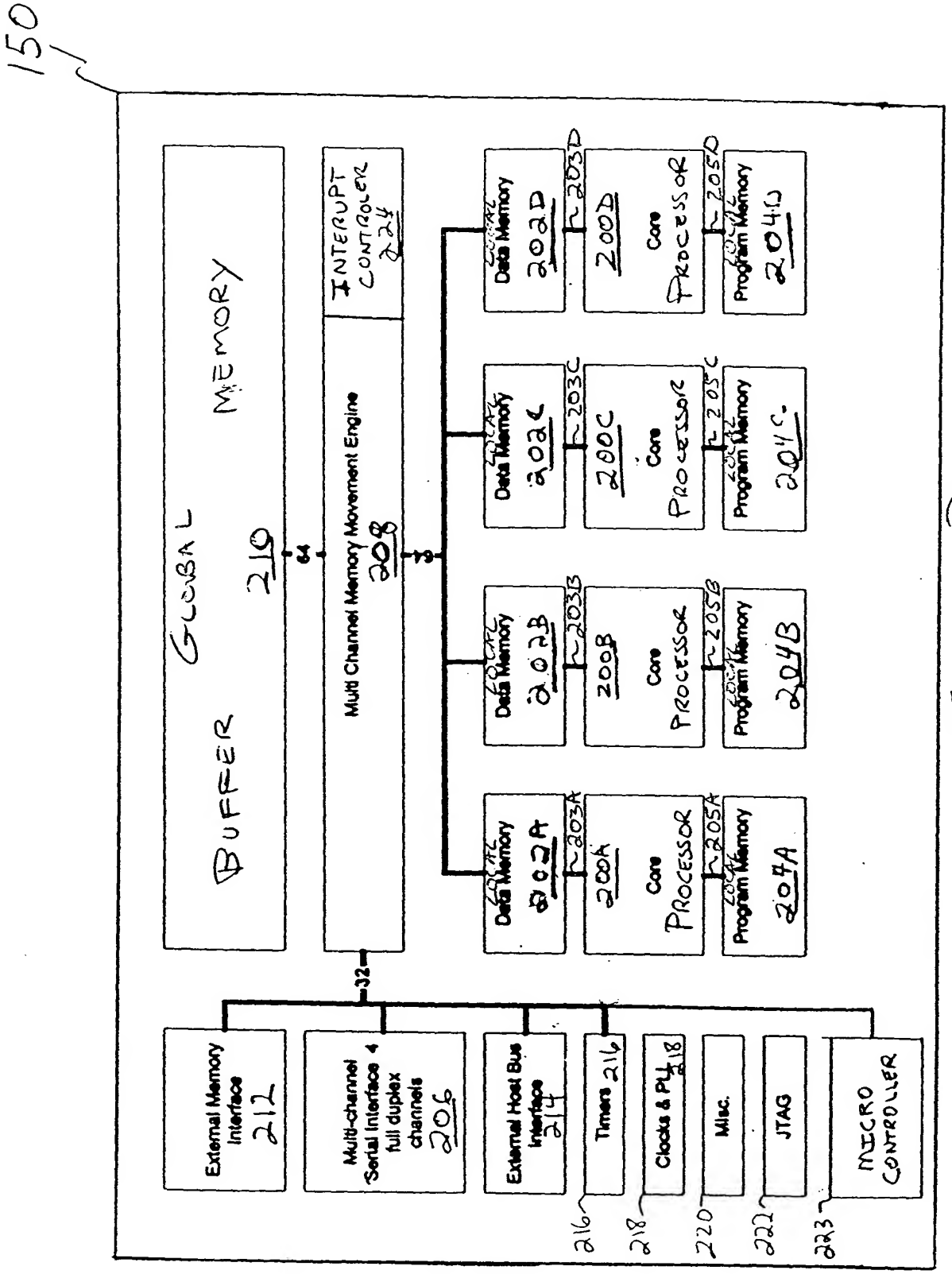

150
GLOBAL MEMORY BUFFER 210
64
INTERRUPT CONTROLLER 224
Multi Channel Memory Movement Engine 208
3
32
Data Memory 202A
Program Memory 204A
Core Processor 200A
203A
205A
Data Memory 202B
Program Memory 204B
Core Processor 200B
203B
205B
Data Memory 202C
Program Memory 204C
Core Processor 200C
203C
205C
Data Memory 202D
Program Memory 204D
Core Processor 200D
203D
205D
External Memory Interface 212
Multi-channel Serial Interface 4 full duplex channels 206
External Host Bus Interface 214
Timers 216
Clocks & PLL 218
Misc. 220
JTAG 222
MICRO CONTROLLER 223

FIG. 2

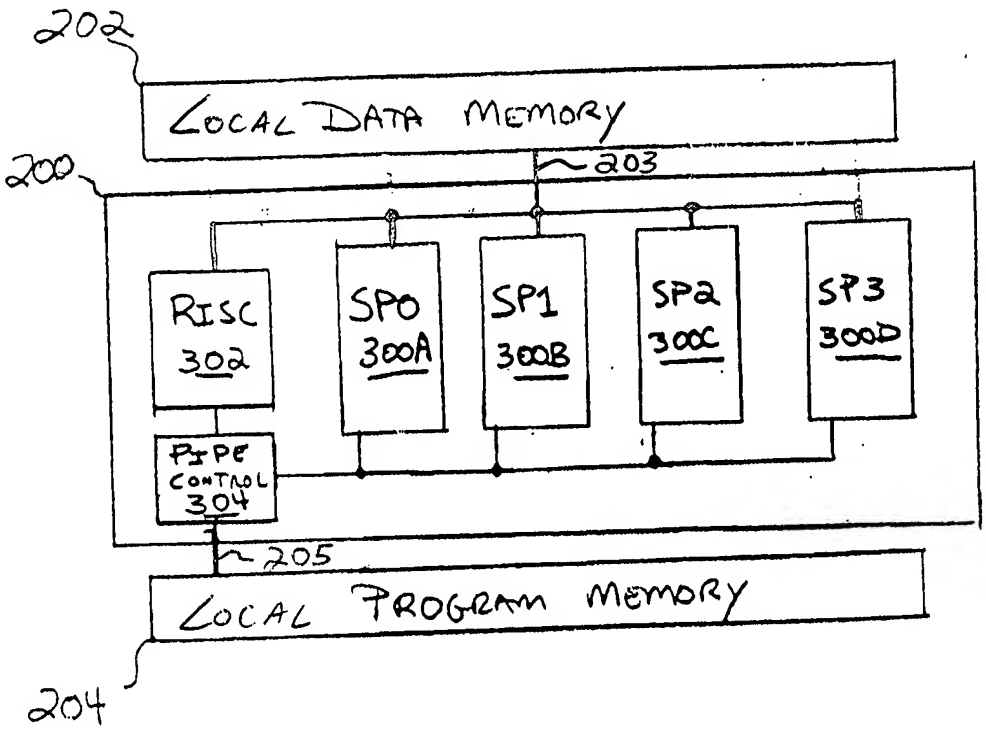


FIG. 3

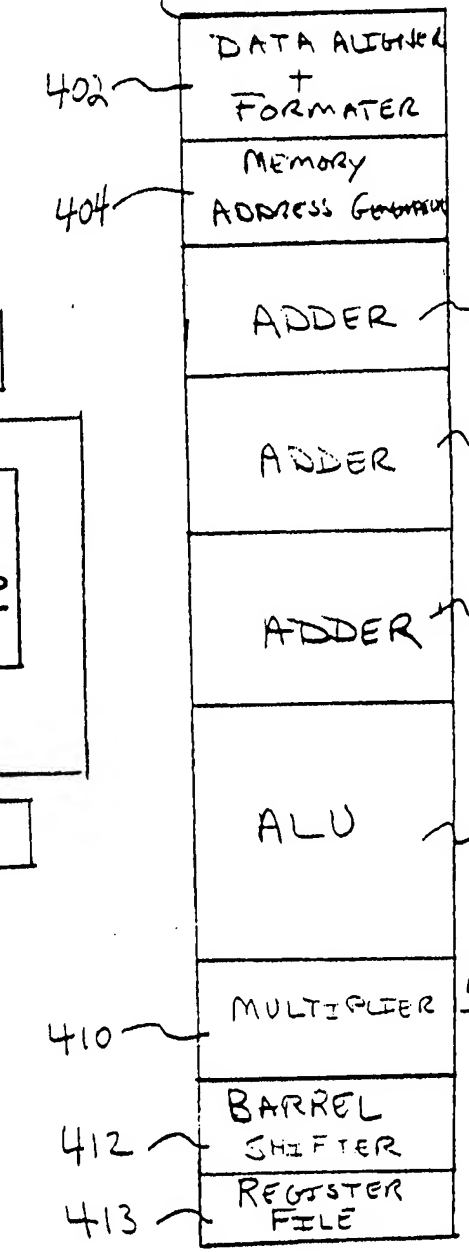


FIG. 4

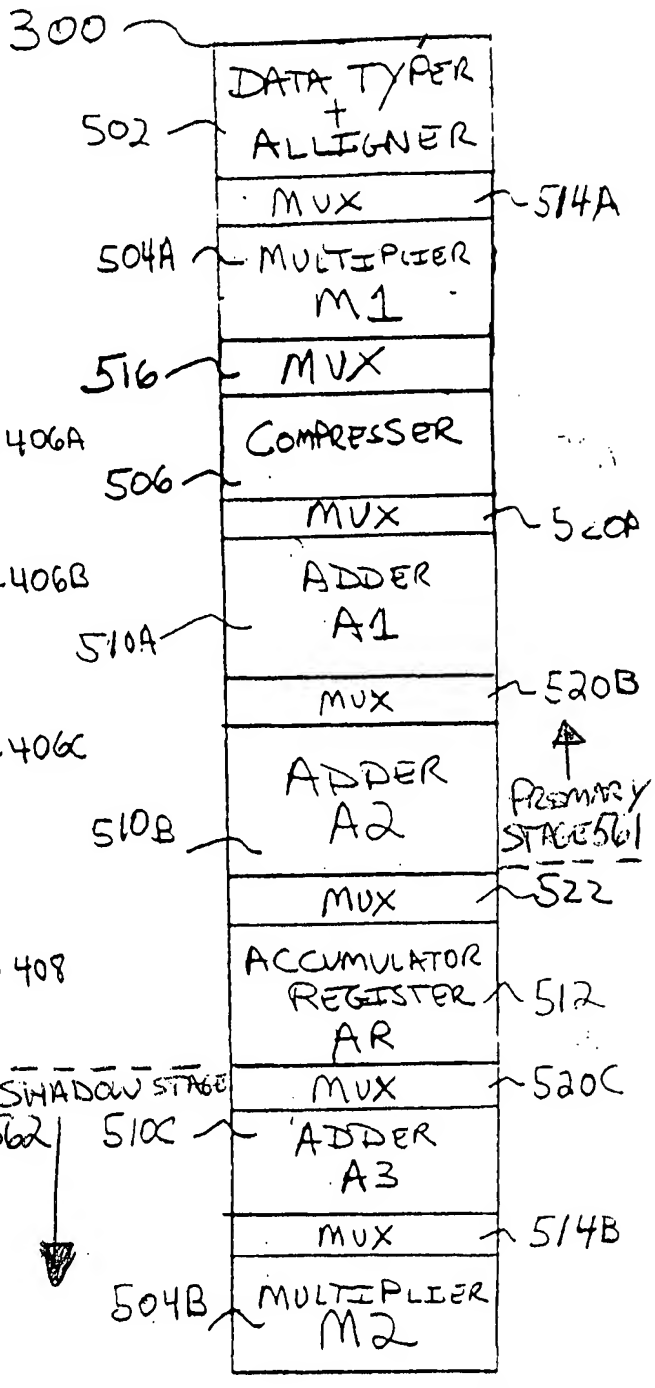


FIG. 5A

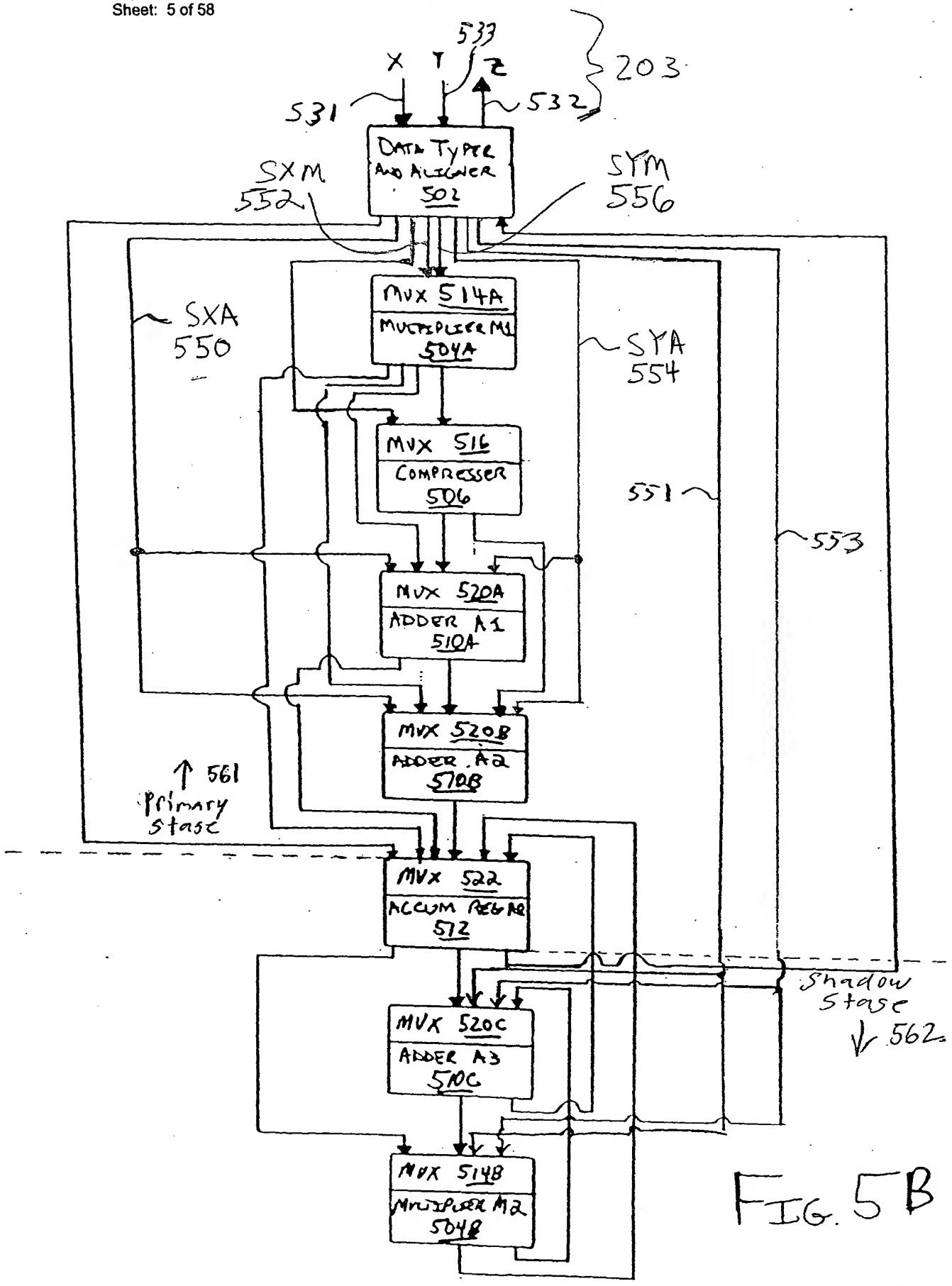

533
X
Y
Z
203
531
532
DATA TYPE AND ALIGNER 502
SXM 552
SYM 556
MUX 514A
MULTIPLIER M1 504A
SXA 550
SYA 554
MUX 516
COMPRESSER 506
551
553
MUX 520A
ADDER A1 510A
MUX 520B
ADDER A2 510B
561
Primary Stage
MUX 522
ACCUM REGR 512
Shadow Stage
562
MUX 520C
ADDER A3 510C
MUX 514B
MULTIPLIER M2 504B

FIG. 5B

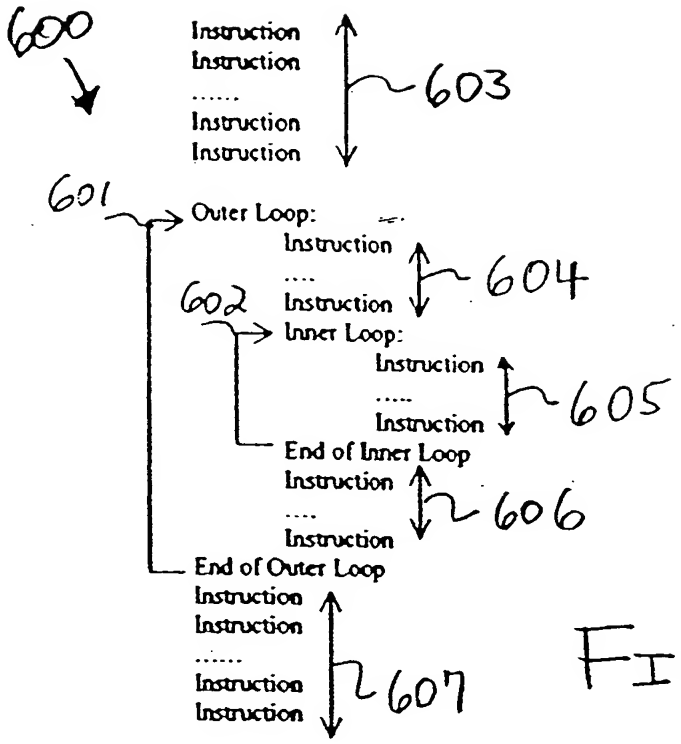


FIG. 6A

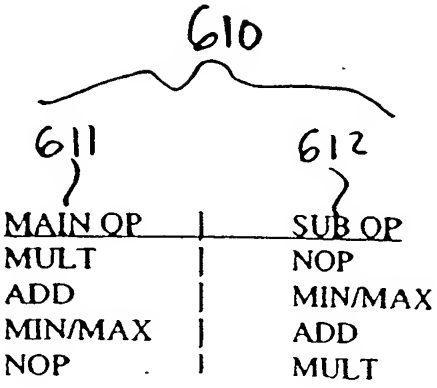


FIG. 6G

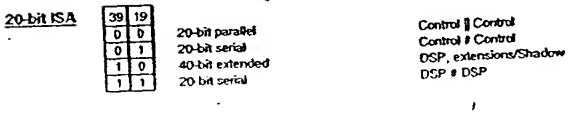


FIG. 6B

6-bit operand specifier:

A 6-bit specifier is used in DSP extended instructions to access memory and register operands.

5	4	3	2	1	0
M/R					

0	0	ac-page	ereg
0	1	gpr: r0-r15	GPR
1	ptr : (r0) to (r15)	off	Mem[ptr[0-15]] || ptr[0-15] += offset1/offset2 Always postupdate

This allows access to data memory, ereg and GPR

- Bit 5 = 1: Use rX (X: 0-7) register to obtain effective memory address and post-modify the ptr field by one of two possible offsets specified in rX registers.
  dmem[ptr], ptr = ptr + offset1, if off = 0
  ptr = ptr + offset2, if off = 1
- Bit 5 = 0: Access ac-page or GPR

If Bit-4 is set to 0, then bits 3:0 control access to the general-purpose register file (r0-15) or to execution unit registers.

GPR	GPR Intr page	ac-page	ac intr page	ereg-Shadow DSP
R0	R0	A0	A0_i	A0
R1	R1	A1	A1_i	A1
R2	R2	T	T	T
R3	R3	TR	TR	TR
R4	R4			
R5	R5			
R6	R6			
R7	R7			
R8	R8			SX1
R9	R9			SX1s
R10	R10			SX2
R11	R11			SX2s
R12	R12_i			SY1
R13	R13_i			SY1s
R14	R14_i			SY2
R15	R15_i			SY2s

FIG. 6C

For shadow DSP instructions, the 3-bit specifier for operands is defined as follows:

2	1	0	
0	0	0	A0
0	0	1	A1
0	1	0	T
0	1	1	TR
1	0	0	SX1
1	0	1	SX1s
1	1	0	SX2
1	1	1	SX2s

EREG1

2	1	0	
0	0	0	A0
0	0	1	A1
0	1	0	T
0	1	1	TR
1	0	0	SY1
1	0	1	SY1s
1	1	0	SY2
1	1	1	SY2s

EREG2

FIG. 6E

Only the shadow DSP instructions can see the above modified page of execution unit registers.

4-bit operand specifier:

Memory operands: (rX) specifies an access out of the data memory to the execution unit for the function that needs to be performed. The address for the access is specified in the rX register in the general register file that hold the 14-bit pointer (16K of addressing) to memory, 5-bit signed offset or a 3-bit unsigned offset that can post-modify the address. In addition each pointer is typed for efficient SIMD processing and includes a permute control for rearranging data elements of a vector on the fly. The "podi" core can deal with 4-element 16-bit real vectors or complex data directly. This ability to manipulate memory data directly reduces the instruction width greatly and allows efficient signal processing.

(rX): Memory Address Registers

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
type				cb	x	permute				off1: (0-7)			off0: (-16 to 15)					ptr: pointer													

FIG. 6D

5-bit operand specifier:

The 5-bit specifier includes the 4-bit specifier for general data operands and the special purpose registers. It is used in RISC instructions.

4	3	2	1	0
0	spr: s0-s15			
1	gpr: r0-r15			

SPR

	SPR
0	fu-ctl
1	a-type
2	ps-ctl
3	t-type
4	pl-ctl
5	cb-ctl
6	shuffle
7	io-ptr
8	status
9	loop-ctl
10	pcr
11	reserved
12	reserved
13	reserved
14	reserved

Intr page

SPR intr page	
fu-ctl_I	
a-type_I	
ps-ctl	
t-type	
pl-ctl	
cb-ctl_I	
shuffle	
io-ptr	
status_I	
loop-ctl	
pcr	stack(8)
reserved	
reserved	
reserved	
reserved	

NOTE: All SPR registers are reset to all zeros at power on reset except for the PCR register.

FIG. 6F

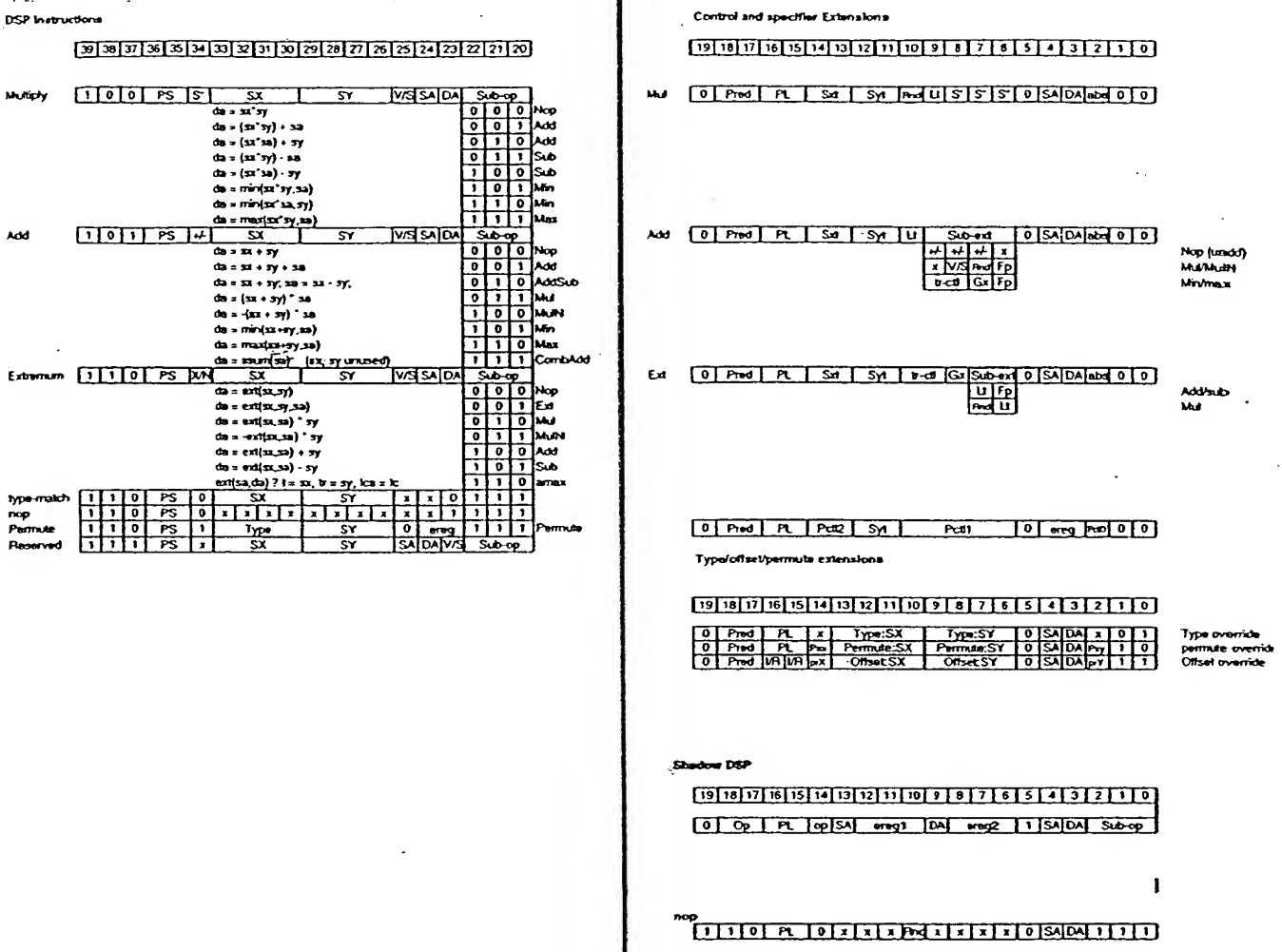

DSP Instructions
Multiply
Add
Extremum
type-match
nop
Permute
Reserved
Control and specifier Extensions
Mul
Add
Ext
Nop (unadd)
Mul/MulN
Min/max
Add/sub
Mul
Type/offset/permute extensions
Type override
permute override
Offset override
Shadow DSP
nop

FIG. 6H

FIG. 6I

Control Instructions

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
add,sub	1	Pred	0	0	0	RX		RY			RZ				+/-	0
max,min	1	Pred	0	0	0	RX		RY			RZ				X/N	1
Shift	1	Pred	0	0	1	RX		U14			RZ				U11	R/L
Logic	1	Pred	0	1	0	RX		RY			RZ				&	&!
Mux	1	Pred	0	1	1	RX		RY			RZ				Pd	0
mov	1	Pred	0	1	1	RX		DZ			Rzn	Dzn	0	0	0	1
add8	1	Pred	0	1	1	SI4		DZ			x	x	1	0	0	1
mov2erg	1	Pred	0	1	1	RX		unit	ereg		gd	N/E	x	x	1	0	1
Ldm	1	Pred	0	1	1	RX		DZ1			DZ2				1	1
Set4bits	1	Pred	1	0	0	U14:POS		RZ			Rzn	U14				0
Set2bits	1	Pred	1	0	0	U14:POS		RZ			Rzn	U12		0	0	1
Setbit	1	Pred	1	0	0	U14:POS		RZ			Rzn	U11	U11	1	0	1
Movi	1	Pred	1	0	0	SI8					RZ				1	1
Jmp	1	Pred	1	0	1	SI9						0	Pred		0	0
Call	1	Pred	1	0	1	SI9						1	Pred		0	0
Loop	1	Pred	1	0	1	U15 : Lcount			U15: Lsize				U12: Lst		0	1
Jmpi	1	Pred	1	0	1	RX	x	x	x	x	x	0	Pred		1	0
Calli	1	Pred	1	0	1	RX	x	x	x	x	x	1	Pred		1	0
Loopi	1	Pred	1	0	1	RX	x	U45: Lsize					U12: Lst		1	1
Test	1	Pred	1	1	0	RX		RY			PZ			=, <, >		0
Testbit	1	Pred	1	1	0	RX		U15			PZ		B		0	1
Andp, orp	1	Pred	1	1	0	Pa		Pb		Pc	PZ		&!		1	1
Load	1	Pred	1	1	1	MX		RZ			Ext				0	0	0
Store	1	Pred	1	1	1	MZ		RX			Ext				1	0	0
eLoad	1	Pred	1	1	1	MX		RZ			1	1	1	0	0	0
eStore	1	Pred	1	1	1	MZ		RX			1	1	1	1	0	0
Extended	1	Pred	1	1	1	Bits 27:16									1	0
Logic2	1	Pred	1	1	1	RX		RY/RZ			Rzn	Ryn	&	&!	0	1
mov-erg	1	Pred	1	1	1	unit	ereg		RZ		gd	SR		0	1	1
Crb	1	Pred	1	1	1	RX		RZ			sbm	0	0	1	1	1
Parity	1	Pred	1	1	1	RX		PZ		O/E	0	1	0	1	1	1
Stm	1	Pred	1	1	1	MZ		RX			1	1	0	1	1	1
Abs	1	Pred	1	1	1	RX		RZ			0	0	1	1	1	1
Neg	1	Pred	1	1	1	RX		RZ			0	1	1	1	1	1
Div-step	1	Pred	1	1	1	RX		RZ			1	0	1	1	1	1
[illegible]	1	Pred	[illegible]													
Reserved	1	Pred	x	x	x	[illegible]										
Return	1	Pred	1	1	1	Pred	I-cll				0	1	0	1	1	1
Zero-ac	1	Pred	1	1	1	ac #					1	1	0	1	1	1
eSync	1	Pred	1	1	1	RZ					0	1	1	1	1	1
Swi	1	Pred	1	1	1	UI3				0	1	1	1	1	1	1
Nop	1	Pred	1	1	1	UI3				1	1	1	1	1	1	1

<Bit1, Bits9-6> = UI5 (Shift Amount)

Bit 5: 0=one ereg, 1=broadcast all four; Bit4: 0=16-bit, 1=32-bit

<Bit3, Bits13-10> = UI5 :POS

FIG. 6J

Extended Control

	Bits 13:2 of upper half (39:20)																										
	13	12	11	10	9	8	7	6	5	4	3	2	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Insert/Extract	RX				RZ				0	0	0	0	0	x	x	0	Rzn	Rzn	I/E	FA	FA	Offset:UI5					Length:UI5					0
																					x	RY					RY				x	
Inserti	U14: length				RZ				0	0	0	1	0	x	x	0	Rzn	U15: Position				Imm10										
Shift	RX				RZ				0	0	0	0	0	rzh	rzl	0	rzh	rzl	D	U/S	1	Shift: UI5					A/L	LI	R/L	0	Fill	1
																					0	nyh	RY									
Rotate	RX				RZ				0	0	0	0	0	x	x	0	x	x	x	x	1	Shift: UI5					1	x	R/L	1	x	1
																					0	nyh	RY									
jmp, call	UI7							U/C	0	0	1	0	0	Pred		0	Fl	UI15														
dloop	U14: outer LC				U14: inner LC				0	0	1	1	0	x	ext	0	UI1	U14: outer Lsize				U14: inner Lsize				uz: 0-Ls	U14: inner Lstart				0
dloopi	RX				RY				0	0	1	1	0	x	ext	0	x	U14: outer Lsize				U14: inner Lsize				uz: 0-Ls	U14: inner Lstart				1
mult	RX				RY				0	1	0	0	0	x	Ims	0	0	rxh	rxl	ryh	ryl	+/-	-/+	RZ			vl	rzh	rzl	s/u	s/u	0
add/sub	RX				RY				0	1	0	0	0	x	x	0	0	rxh	rxl	ryh	ryl	+/-		RZ			LI	rzh	rzl	x	x	1
Reserved	[illegible]																															
logicp	PX			D	PZ				0	1	0	0	0	x	x	0	1	T/F	T/F	T/F	&!	&!	PY				PY				x	1
Testi	RX			D	PZ				0	1	0	1	0	=,>,<		0		Imm16														
Movi	HWI	Fill			RZ				0	1	1	0	0	x	x	0		Imm16														
loadi	Type				RZ				0	1	1	1	0	x	x	0	0	0		Imm14													
storei	Type				RX				0	1	1	1	0	x	x	0	0	1		Imm14													
loadli	RX				RZ				0	1	1	1	0	x	x	0	1	Rzn	0	Type		SI10											
storel	MZ				RX				0	1	1	1	0	x	x	0	1	Rzn	1	Type		SI10											
Addi/subi	RX				RZ				1	0	+/-	0	0	LI	s/u	0		Imm16														
mini,maxi	RX				RZ				1	0	X/N	1	0	x	x	0		Imm16														
andi,ori	RX				RZ				1	1	&!	R/L	0	x	x	0		Imm16														

Fill: Sign/Zero

Fl = PC relative
Bit 15 is continuation of inner LC

andp, orp, andorp, orandp: pz = (px relop py) relop pv)

FIG. 6K

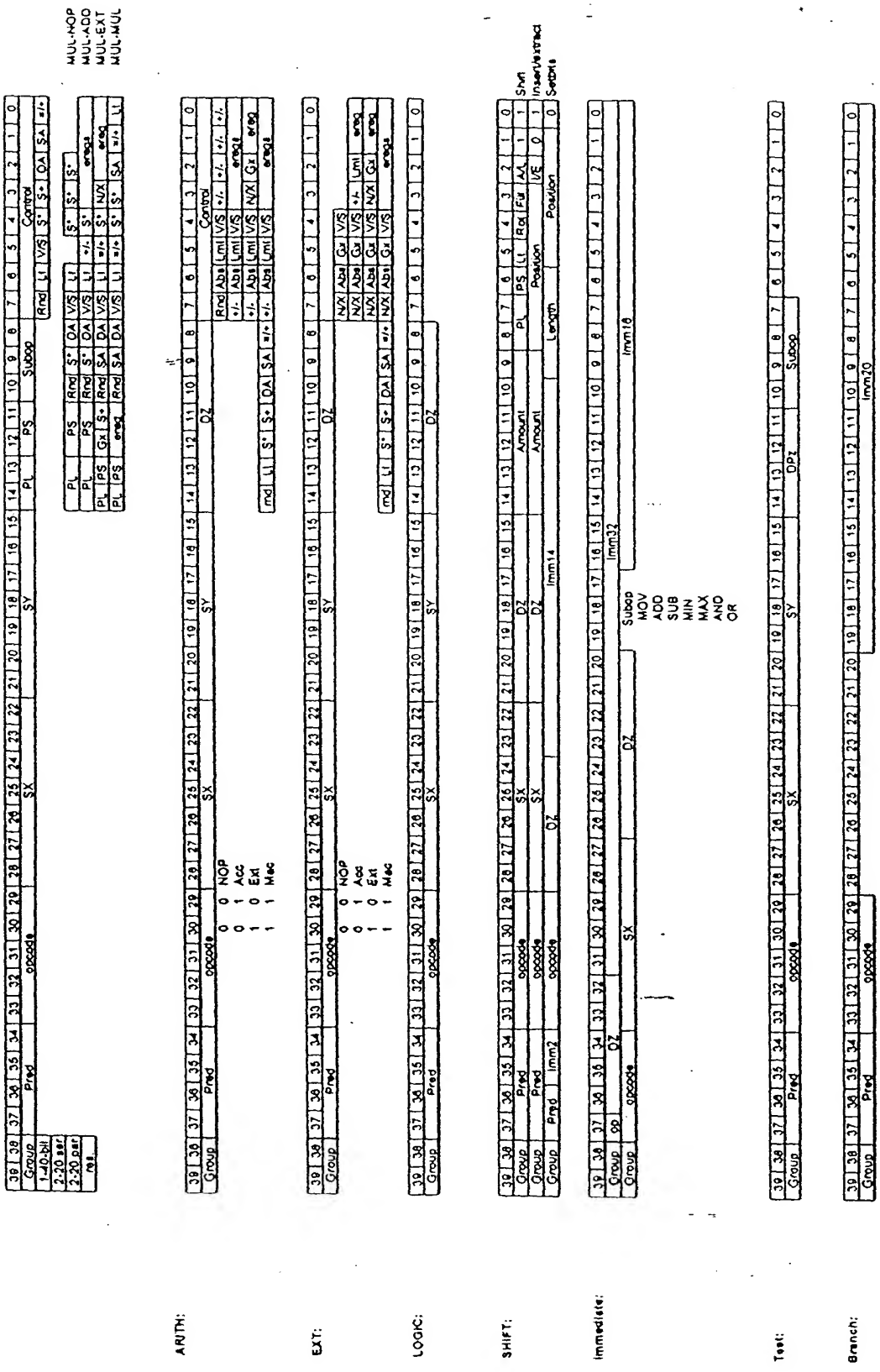

MAC:
ARITH:
EXT:
LOGIC:
SHIFT:
Immediate:
Test:
Branch:
Misc:
MUL-NOP
MUL-ADD
MUL-EXT
MUL-MUL
NOP
Acc
Ext
Mac
Subop
MOV
ADD
SUB
MIN
MAX
AND
OR

FIG. 6L

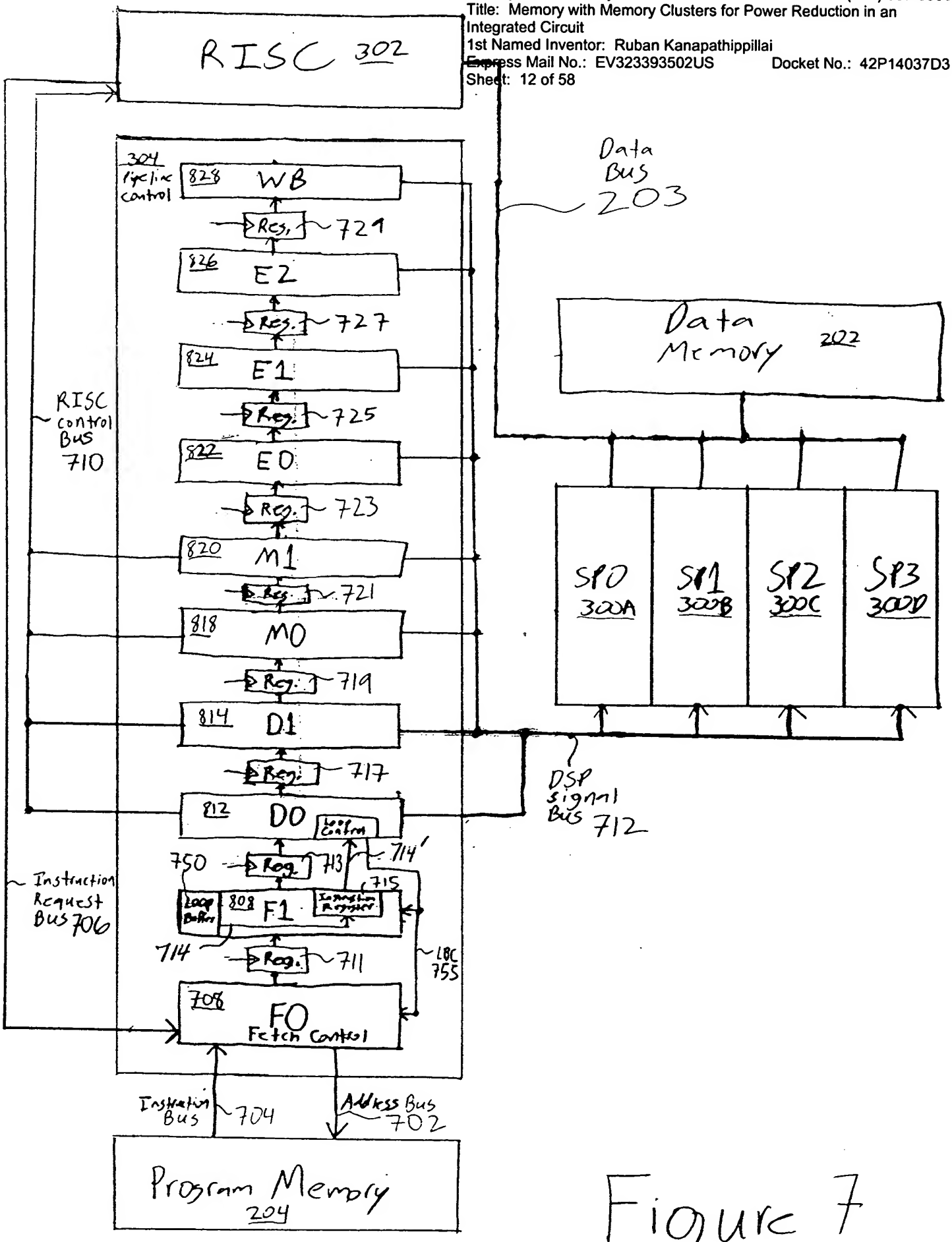

RISC 302
304 cyclic control
828 WB
Reg. 721
826 E2
Reg. 727
824 E1
Reg. 725
822 E0
Reg. 723
820 M1
Reg. 721
818 M0
Reg. 719
814 D1
Reg. 717
812 D0
Reg. 713
714'
750
715
808 F1
714
Reg. 711
LBC 755
708 F0 Fetch Control
RISC control Bus 710
Instruction Request Bus 706
Data Bus 203
Data Memory 202
SP0 300A
SP1 300B
SP2 300C
SP3 300D
DSP signal Bus 712
Instruction Bus 704
Address Bus 702
Program Memory 204

Figure 7

Pipeline Controller 304

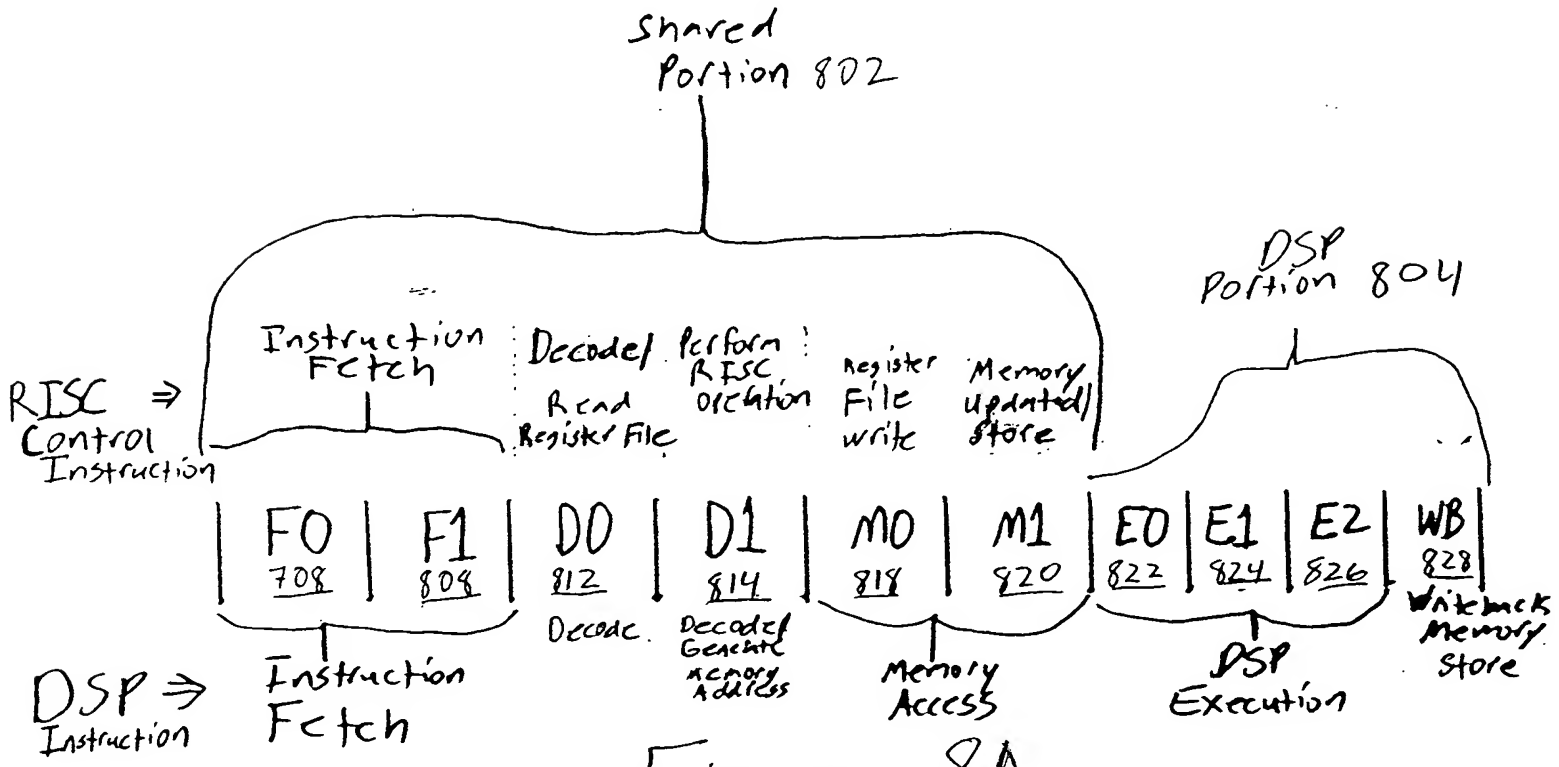


Figure 8A

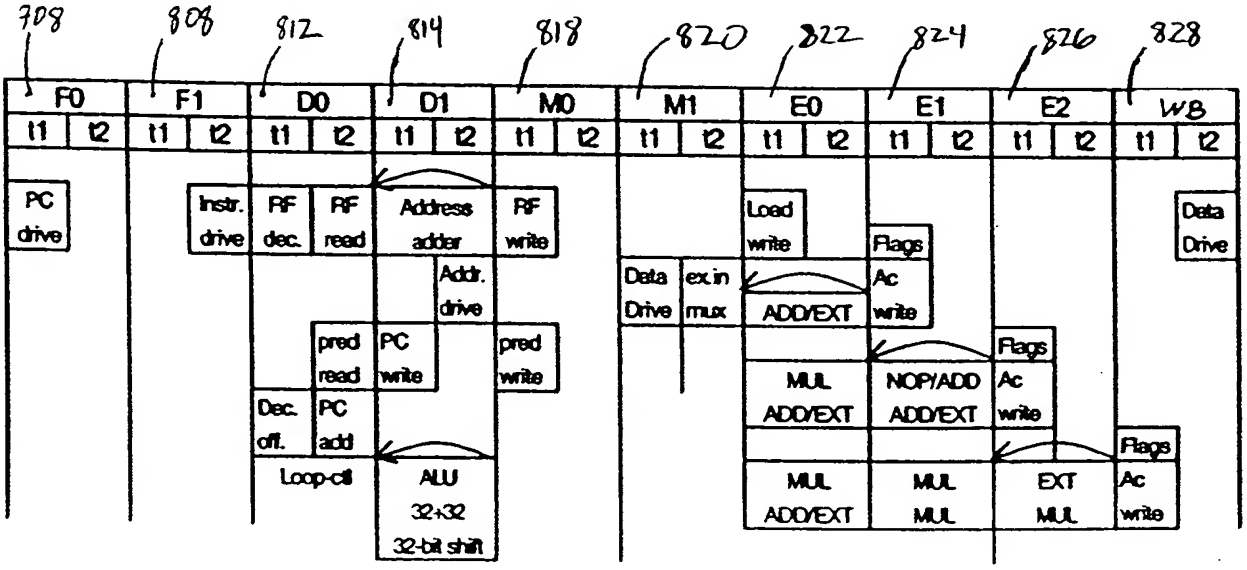


Figure 8B

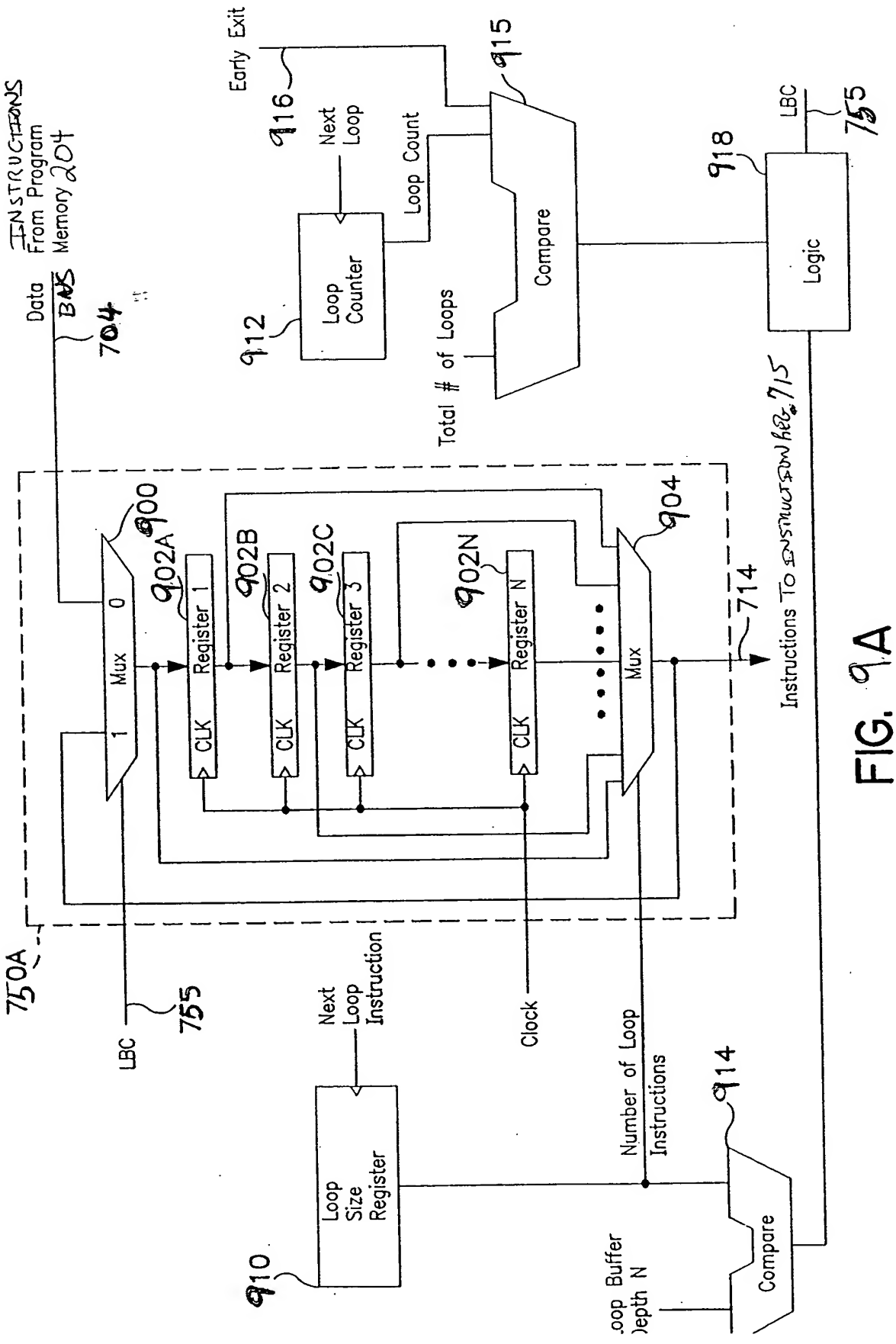

Instructions From Program Memory 204
Data Bus 704
Early Exit
916
Next Loop
Loop Count
915
Loop Counter
912
Total # of Loops
Compare
918
Logic
LBC
755
Instructions To Instruction Mux 715
900
902A
902B
902C
902N
Register 1
Register 2
Register 3
Register N
CLK
Mux
904
714
750A
LBC
755
Next Loop Instruction
Clock
Number of Loop Instructions
914
Loop Size Register
910
Loop Buffer Depth N
Compare

FIG. 9A

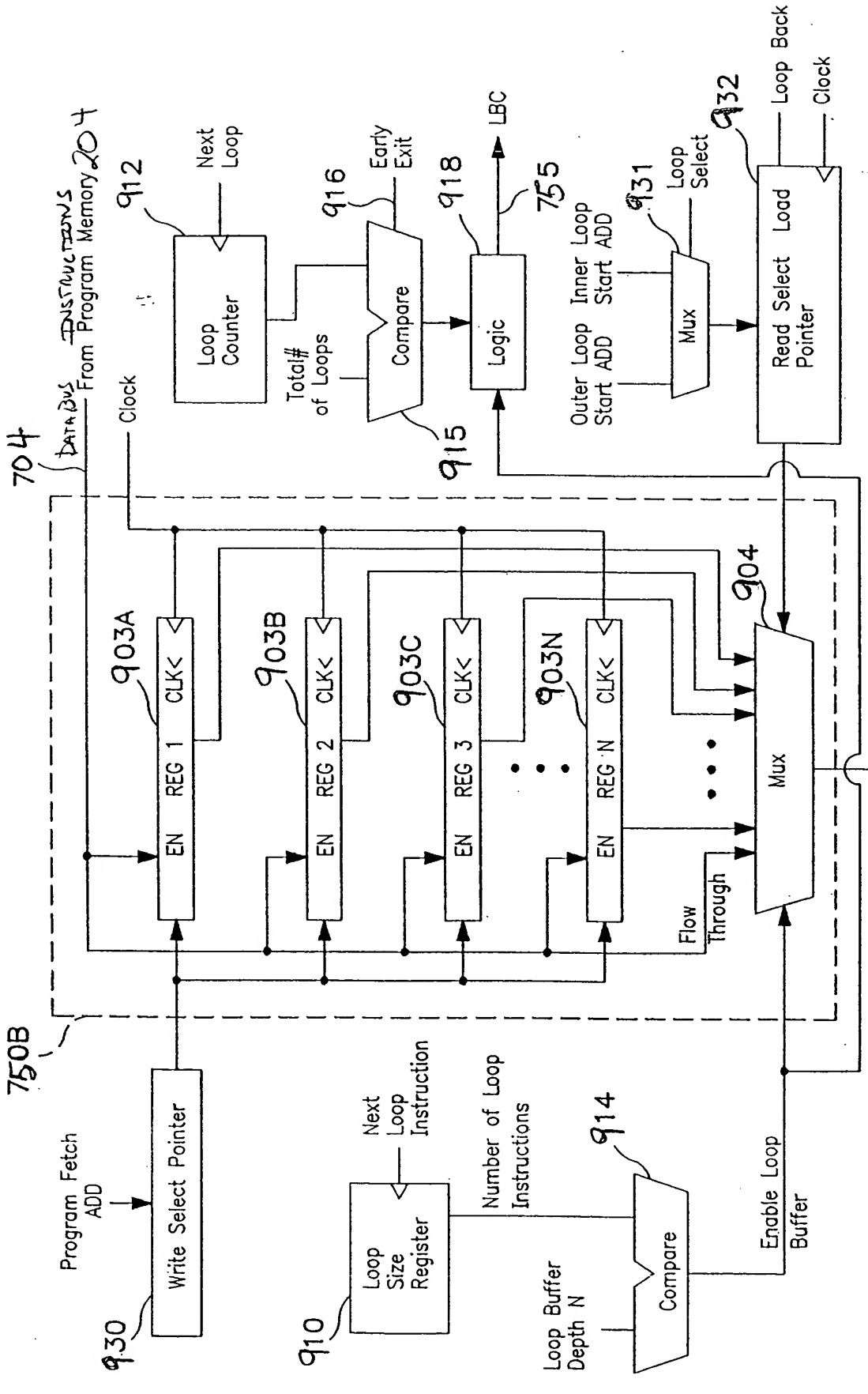

704
Data Bus
Instructions From Program Memory 204
Clock
912
Next Loop
Loop Counter
916
Early Exit
Total# of Loops
Compare
915
918
LBC
755
Logic
931
Loop Select
Outer Loop Start ADD
Inner Loop Start ADD
Mux
932
Loop Back
Clock
Read Select Load Pointer
903A
903B
903C
903N
EN REG 1 CLK<
EN REG 2 CLK<
EN REG 3 CLK<
EN REG N CLK<
904
Mux
714
Instructions TO INSTRUCTION REGISTER 715
Flow Through
750B
Program Fetch ADD
Write Select Pointer
930
Next Loop Instruction
Loop Size Register
910
Number of Loop Instructions
914
Compare
Loop Buffer Depth N
Enable Loop Buffer

FIG. 9B

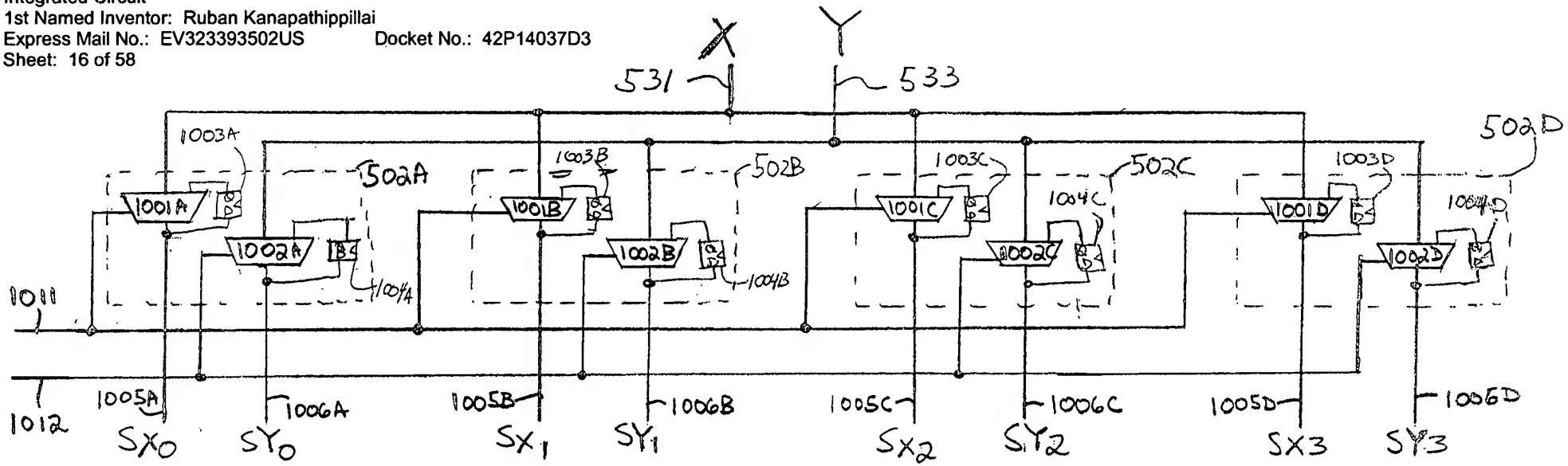

X
Y
531
533
502A
502B
502C
502D
1003A
1003B
1003C
1003D
1001A
1001B
1001C
1001D
1002A
1002B
1002C
1002D
1004A
1004B
1004C
1004D
1011
1012
1005A
1005B
1005C
1005D
1006A
1006B
1006C
1006D
SX0
SY0
SX1
SY1
SX2
SY2
SX3
SY3

FIG. 10

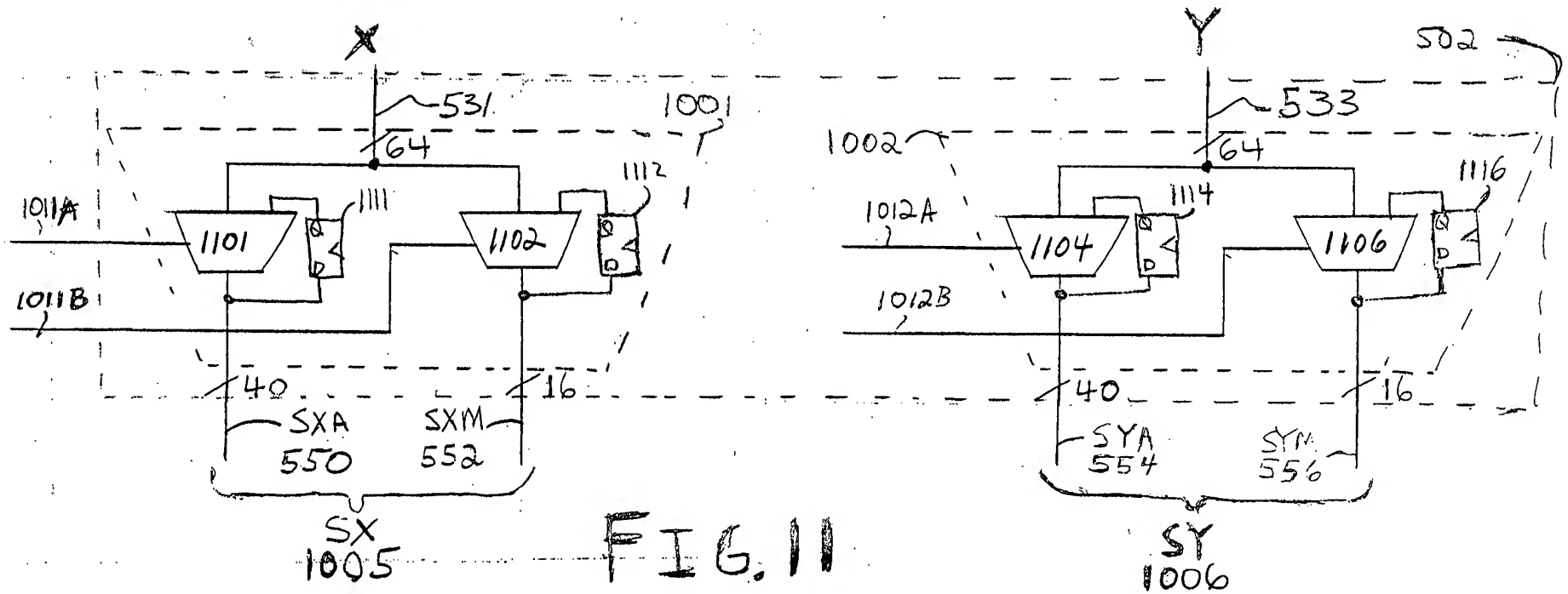
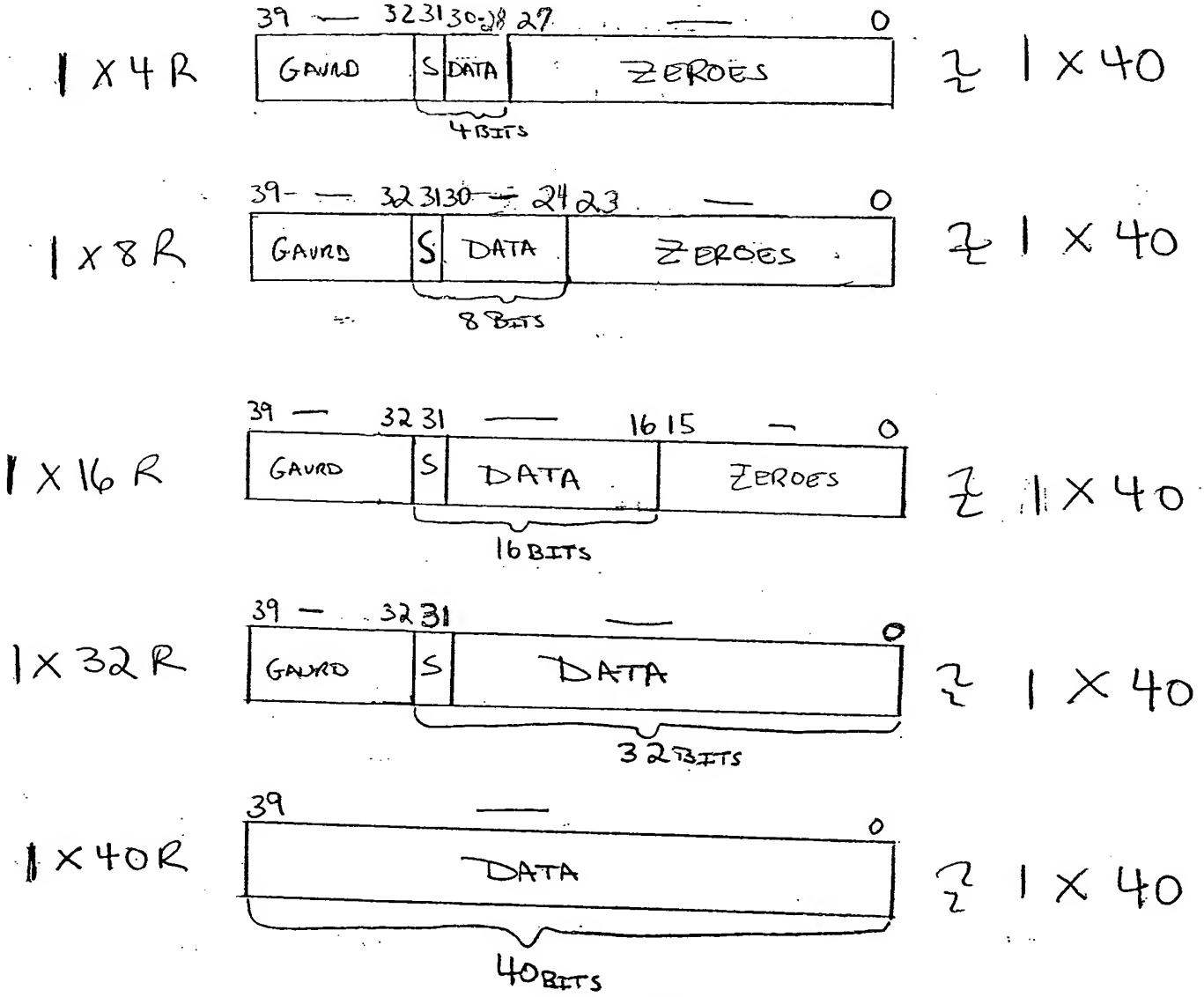

X
Y
531
533
1001
1002
502
64
64
1011A
1011B
1012A
1012B
1101
1102
1104
1106
1111
1112
1114
1116
40
16
40
16
SXA
550
SXM
552
SYA
554
SYM
556
SX
1005
SY
1006

FIG. 11

DATA TYPE

SP CONFIGURATION

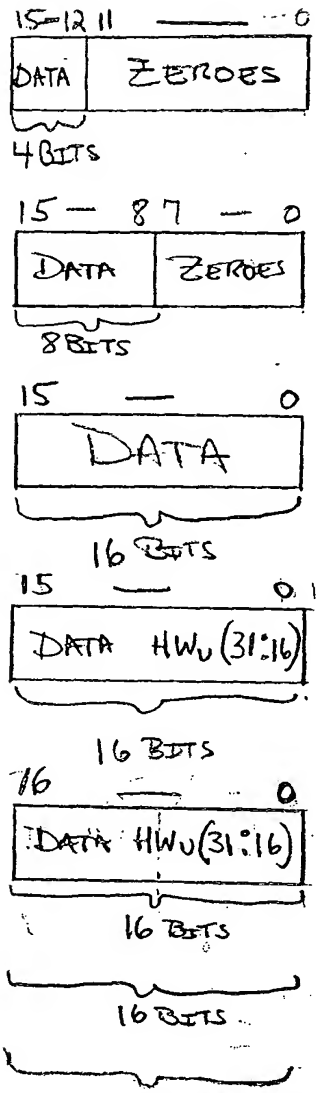


SXA 550 OR SYA 554

FIG. 12A

DATA TYPE

SP CONFIGURATION



SXM 552A-552B
OR
SYM 556A-556B

FIG. 12B

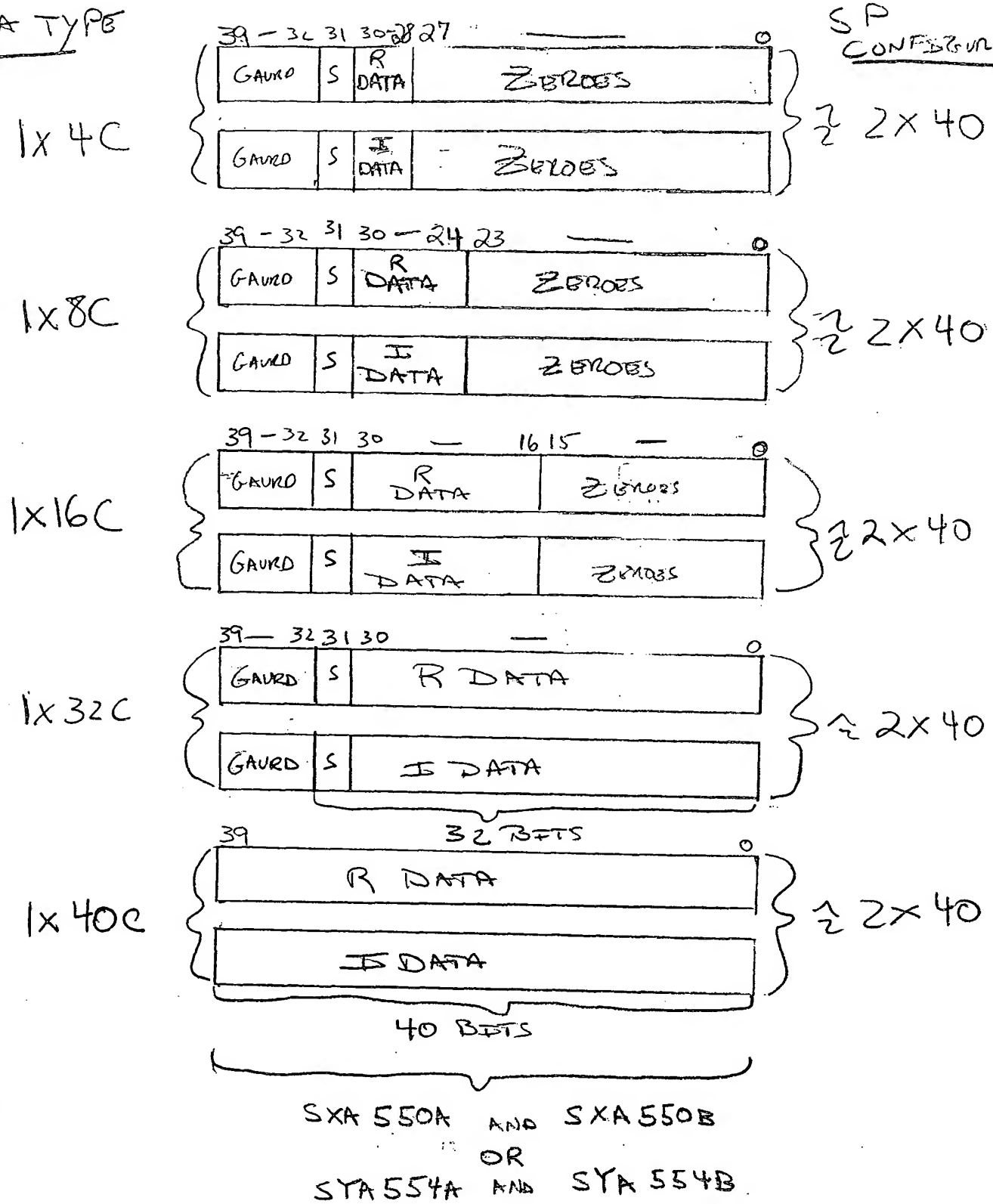


FIG. 12C

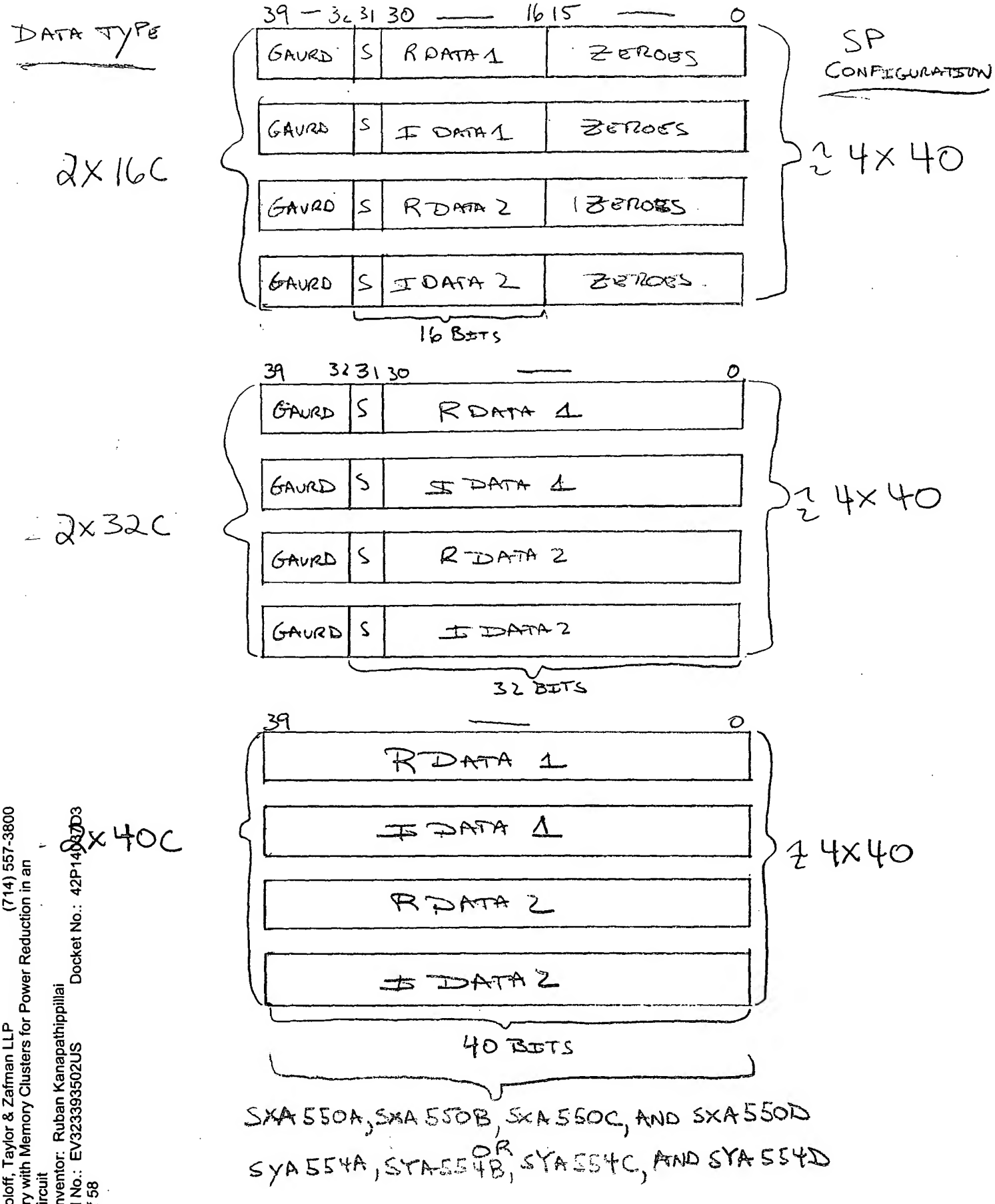
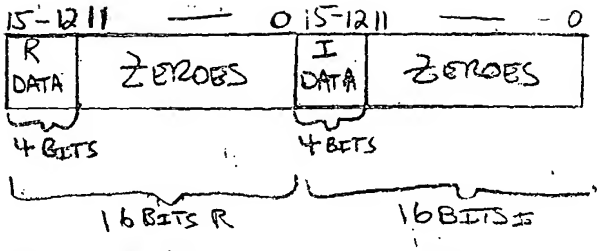


FIG. 12D

DATA TYPE

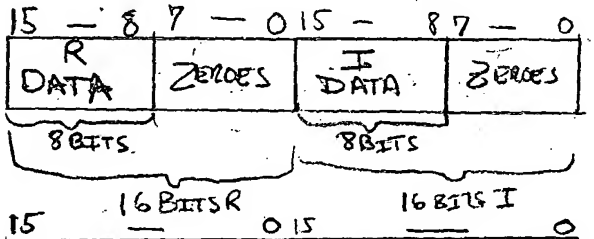
SP CONFIGURATION

1X4C



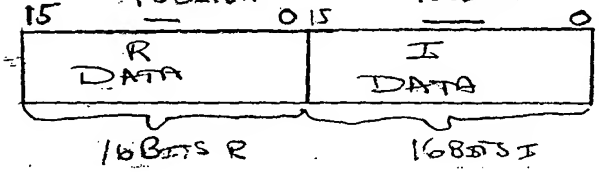
½ 2X16

1X8C



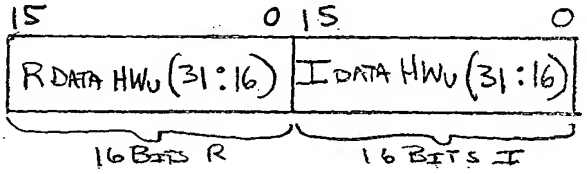
½ 2X16

1X16C



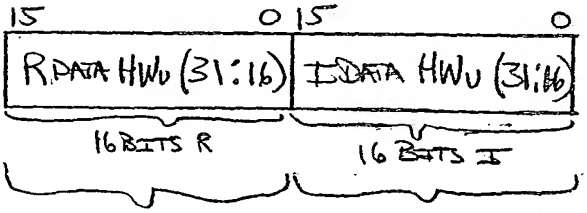
½ 2X16

1X32C



½ 2X16

1X40C



½ 2X16

SXM552A AND SXM552B

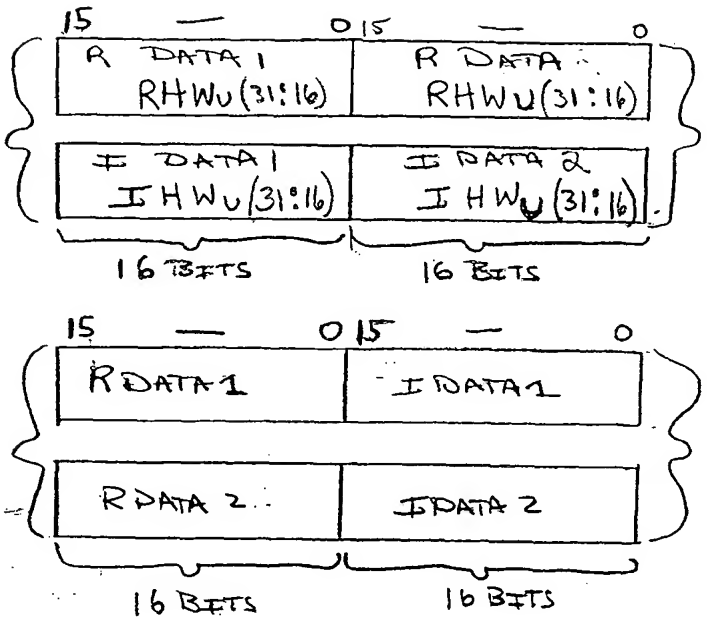
OR

SYM556A AND SYM556B

FIG. 12E

DATA TYPE

SP CONFIGURATION



2 X 32C OR 2 X 40C

≈ 2 X 16

2 1X16C

≈ 2 X 16

SXM552A, SxM552B, SXM552C, AND SXM552D

SYM556A, SYM556B OR, SYM556C, AND SYM556D

FIG. 12F

Operand 1 Data Type:	$N_1 \times S_1$ R
Operand 2 Data Type:	$N_2 \times S_2$ R
Type Matching R:	Max (N_1 or N_2) x Max (S_1 or S_2) R

Fig. 13A

Operand 1 Data Type:	$N_1 \times S_1$ C
Operand 2 Data Type:	$N_2 \times S_2$ C
Type Matching C:	Max (N_1 or N_2) x Max (S_1 or S_2) C

Fig. 13B

Operand 1 Data Type:	$N_1 \times S_1$ R
Operand 2 Data Type:	$N_2 \times S_2$ C
Type Matching R+C:	Max (N_1 or N_2) x Max (S_1 or S_2) C

Fig. 13C

	1x16 real	2x16 real	1x16 cmpx	4x16 real	2x16 cmpx	1x32 real	2x32 real	1x32 cmpx	4x32 real	2x32 cmpx	1x40 real	2x40 real	1x40 cmpx	4x40 real	2x40 cmpx
1x16 real	1 unit	2 unit	2 unit	4 unit	4 unit	2 unit	4 unit	4 unit							
2x16 real	2 unit	2 unit													
1x16 cmpx	2 unit		4 unit												
4x16 real	4 unit			4 unit											
2x16 cmpx	4 unit														
1x32 real	2 unit														
2x32 real	4 unit														
1x32 cmpx	4 unit														
4x32 real															
2x32 cmpx															
1x40 real															
2x40 real															
1x40 cmpx															
4x40 real															
2x40 cmpx															

FIG. 14

	1x16 real	2x16 real	1x16 cmpx	4x16 real	2x16 cmpx	1x32 real	2x32 real	1x32 cmpx	4x32 real	2x32 cmpx	1x40 real	2x40 real	1x40 cmpx	4x40 real	2x40 cmpx
1x16 real	1 unit	2 unit		4 unit		1 unit	4 unit		4 unit		1 unit	2 unit		4 unit	
2x16 real	2 unit	2 unit				2 unit	2 unit					2 unit			
1x16 cmpx															
4x16 real	4 unit			16 unit		4 unit			4 unit					4 unit	
2x16 cmpx															
1x32 real	1 unit	2 unit		4 unit		1unit	2 unit		4 unit		1 unit	2 unit		4 unit	
2x32 real	4 unit	2 unit				2 unit	2 unit					2 unit			
1x32 cmpx															
4x32 real	4 unit			4 unit		4 unit			4 unit		4 unit			4 unit	
2x32 cmpx															
1x40 real	1 unit					1 unit			4 unit		1 unit				
2x40 real	2 unit	2 unit				2 unit	2 unit					2 unit			
1x40 cmpx															
4x40 real	4 unit		4 unit			4 unit			4 unit					4 unit	
2x40 cmpx															

FIG. 15A

	1x16 real	2x16 real	1x16 cmpx	4x16 real	2x16 cmpx	1x32 real	2x32 real	1x32 cmpx	4x32 real	2x32 cmpx	1x40 real	2x40 real	1x40 cmpx	4x40 real	2x40 cmpx
1x16 real	1 unit	2 unit	2 unit	4 unit	4 unit	1 unit	2 unit	2 unit	4 unit	4 unit	1 unit	2 unit	2 unit	4 unit	4 unit
2x16 real	2unit	2 unit				2 unit	2 unit					2 unit			
1x16 cmpx	2unit		2 unit					2 unit			2 unit		2 unit		
4x16 real	4unit			1 unit		4 unit			4 unit					4 unit	
2x16 cmpx	4unit				4 unit					4 unit					4 unit
1x32 real	1unit	2unit				1unit	2 unit	2 unit	4 unit		1 unit	2 unit	2 unit	4 unit	
2x32 real	2unit	2unit				2unit	2 unit					2 unit			
1x32 cmpx	2unit		2unit			2unit		2 unit			2 unit		2 unit		
4x32 real	4unit			4unit		4unit			4 unit		4 unit			4 unit	
2x32 cmpx	4unit				4unit					4 unit					4 unit
1x40 real	1unit		2unit			1unit		2unit	4unit		1 unit	2 unit		4 unit	
2x40 real	2unit	2unit				2unit	2unit				2unit	2 unit			
1x40 cmpx	2unit		2unit			2unit		2unit					2 unit		
4x40 real	4unit			4unit		4unit			4unit		4unit			4 unit	
2x40 cmpx	4unit				4unit					4unit					4 unit

FIG. 15B

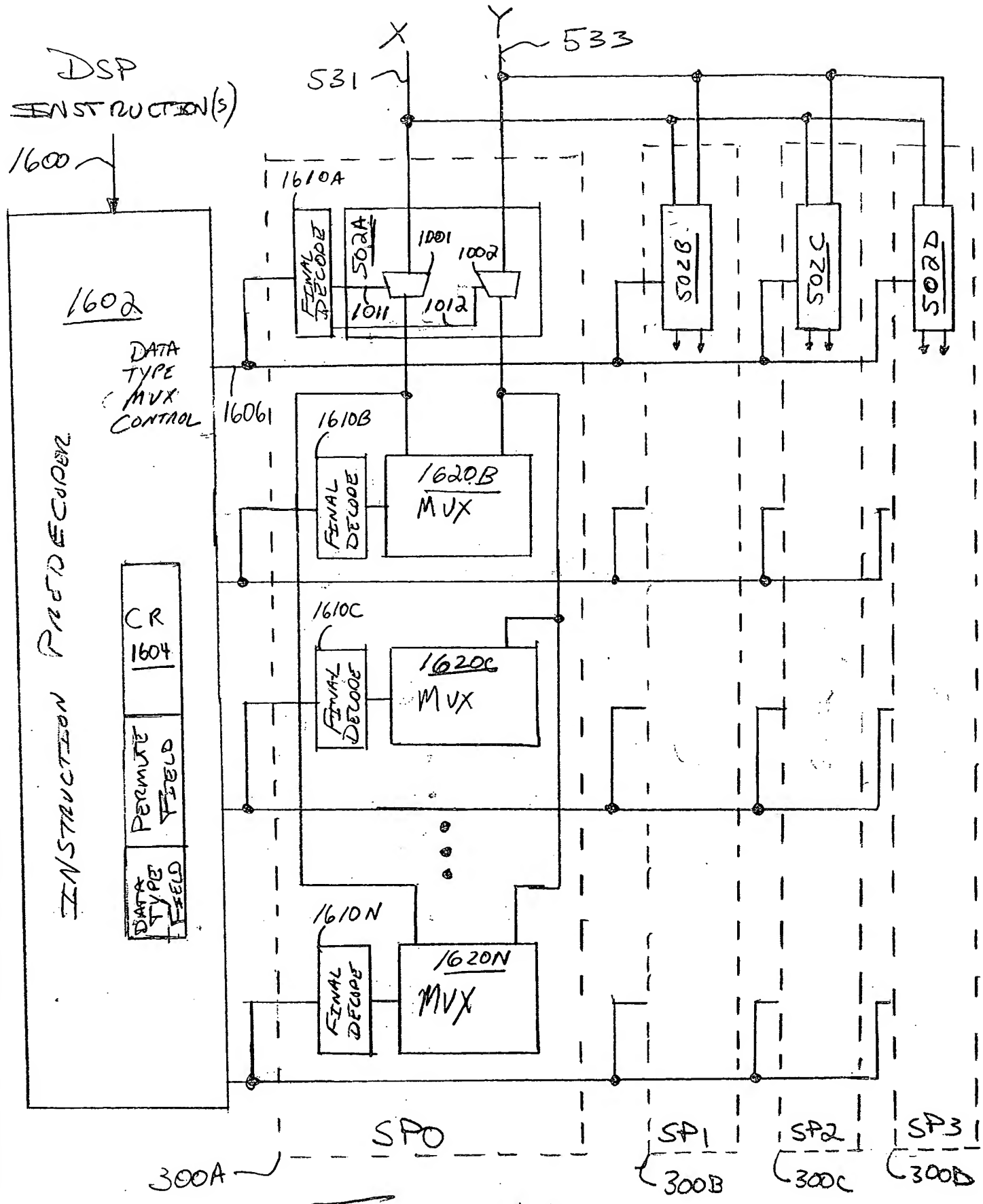

DSP
INSTRUCTION(S)
1600
X
Y
533
531
1610A
FINAL DECODE
502A
1001
1002
1011
1012
1602
DATA TYPE MUX CONTROL
16061
INSTRUCTION PREDECODE
CR 1604
PERMUTE FIELD
DATA TYPE FIELD
1610B
FINAL DECODE
1620B MUX
1610C
FINAL DECODE
1620C MUX
1610N
FINAL DECODE
1620N MUX
502B
502C
502D
SP0
SP1
SP2
SP3
300A
300B
300C
300D

FIG. 16

Data Type: N x S(R/C)

FIG. 17

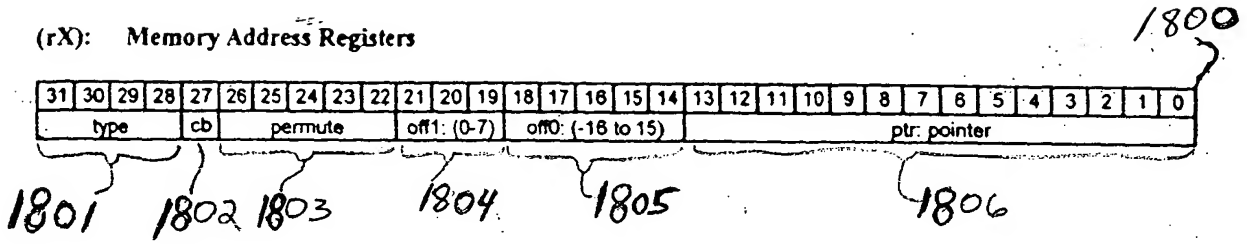


FIG. 18

DATA TYPE 1801

0000: 1x16 real
0001: 2x16 real
0010: 1x16 complex
0011: 4x16 real
0100: 1x32 real
0101: 2x32 real
0110: 1x32 complex
0111: 2X16 complex
1000: 4x32 real
1001: 2x32 complex
1010: 1x40 real
1011: 2x40 real
1100: 1x40 complex
1101: 4x40 real (only for local add unit operations)
1110: 2x40 complex (only for local add unit operations)
1111: Reserved

FIG. 19

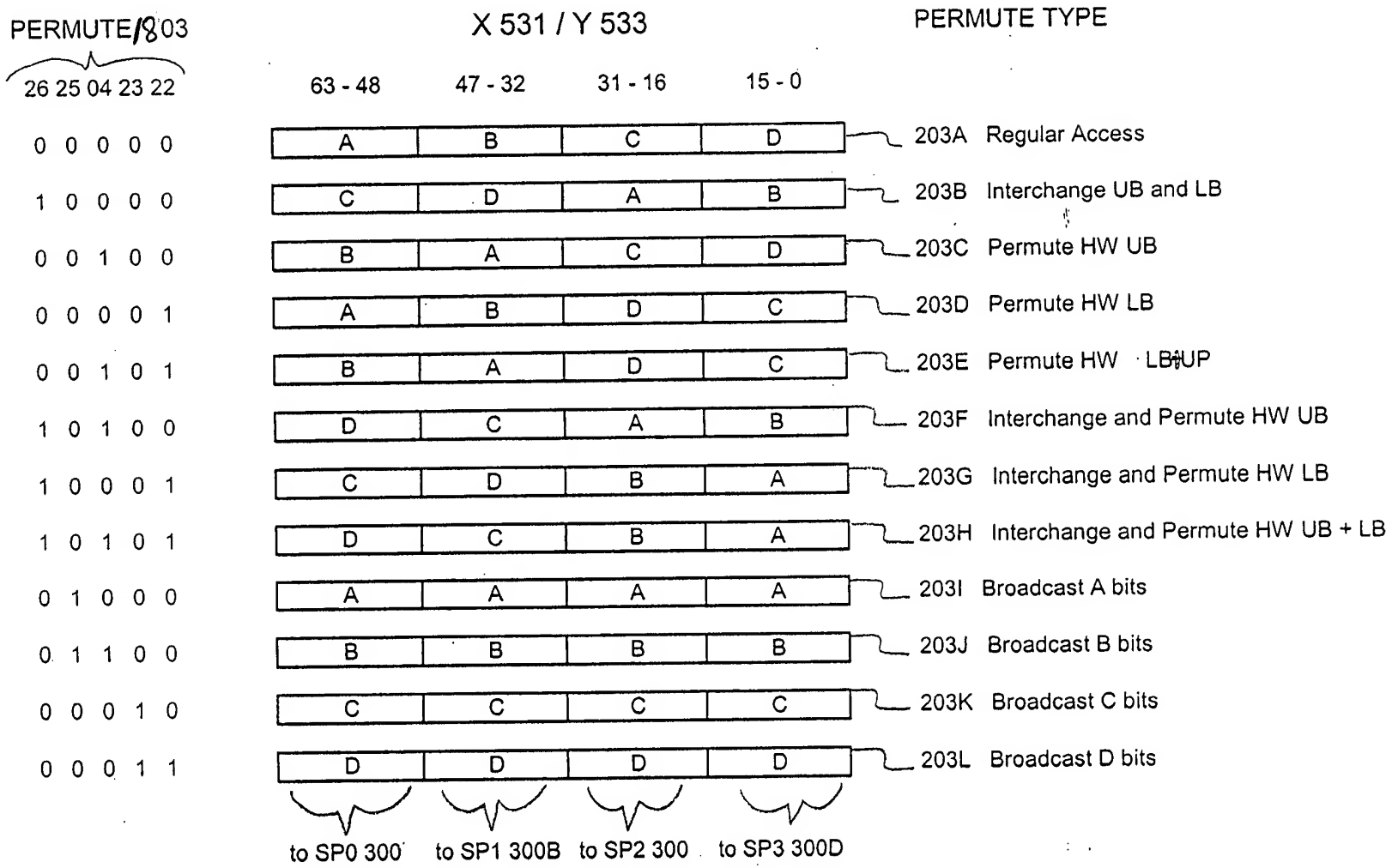


FIG. 20

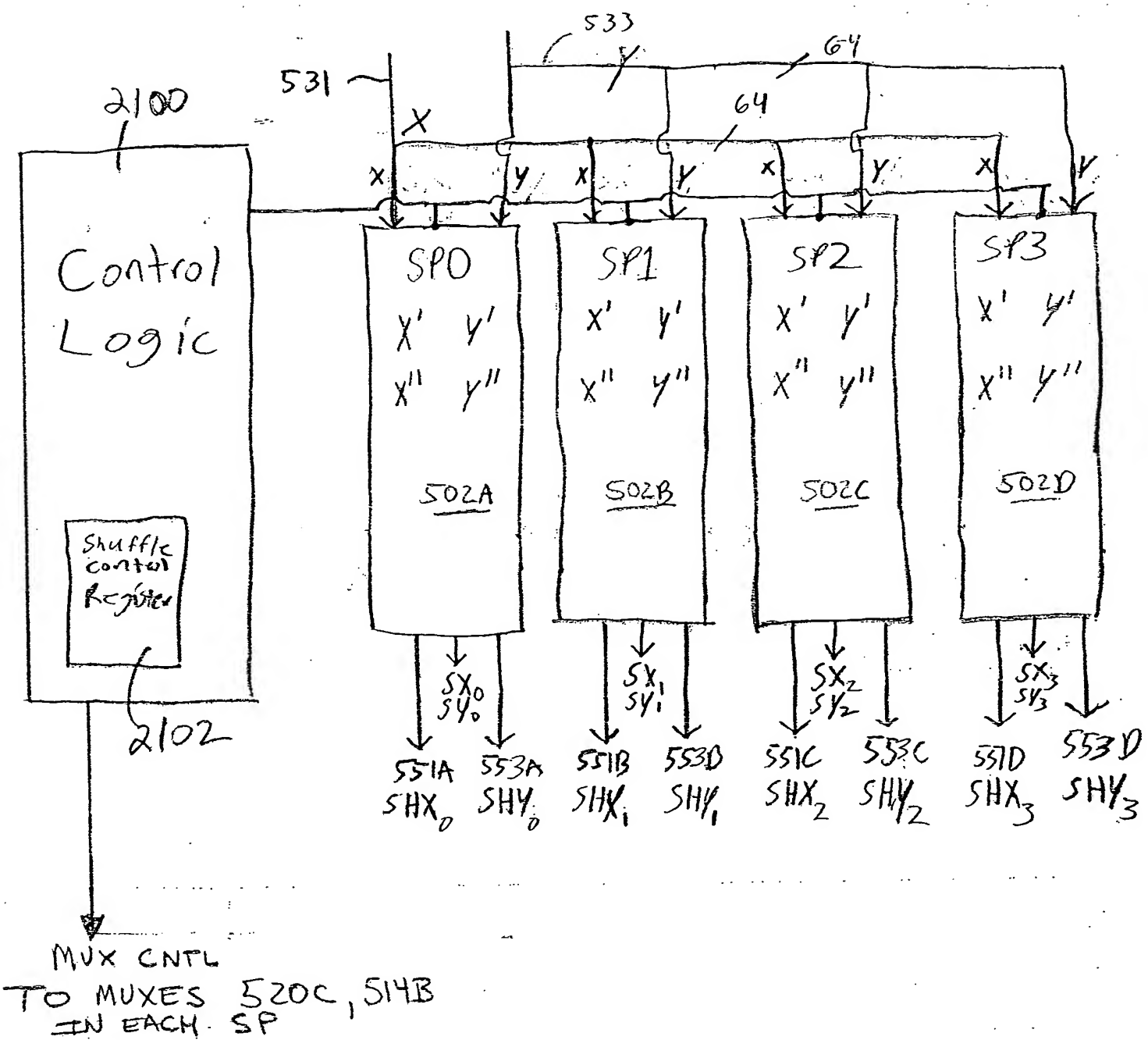

533
64
531
64
2100
X
x
y
Control Logic
Shuffle control Register
2102
SP0
SP1
SP2
SP3
x' y'
x'' y''
502A
502B
502C
502D
SX0 SY0
SX1 SY1
SX2 SY2
SX3 SY3
551A
553A
551B
553B
551C
553C
551D
553D
SHX0
SHY0
SHX1
SHY1
SHX2
SHY2
SHX3
SHY3
MUX CNTL
TO MUXES 520C, 514B IN EACH SP

FIG. 21

$X' = [SX_{10}, SX_{11}, SX_{12}, SX_{13}]$ e.g. $[X_0, X_1, X_2, X_3]$

$X'' = [SX_{20}, SX_{21}, SX_{22}, SX_{23}]$ e.g. $[X_4, X_5, X_6, X_7$

Where SX_{ab} : S=Source; a=delay; b= SP unit number (e.g. SP3, SP2, SP1, SP0; or termed U3, U2, U1, U0,

$y' = [Sy_{10}, Sy_{11}, Sy_{12}, Sy_{13}]$

$y'' = [Sy_{20}, Sy_{21}, Sy_{22}, Sy_{23}]$

Where Sy_{ab} : S= Source; a = delay; b= SP unit number (e.g. SP3, SP2, SP1, SP0; or termed U_3, U_1, U_0)

FIG. 22A

shuffle **Shuffle Control Register**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
u3		u2		u1		u0		u3		u2		u1		u0		u3		u2		u1		u0		u3		u2		u1		u0	
SY2S								SY1S								SX2S								SX1S							

Units are connected to their nearest neighbors for shuffling the sources using the following bit diagram:

00 Unit N+1, SX1 = X' (right)
01 Unit N+1, SX2 = X'' (right)
10 Unit N-1, SX1 = X' (left)
11 Unit N-1, SX2 = X'' (left)

For example to shift the sources to the left by one:

3	2	1	0	From
2	1	0	3	Into

The bits should be 10101010 ($AA)

FIG. 22C

FIR Filter $\begin{bmatrix} X_0 \\ X_1 \\ \vdots \\ X_N \end{bmatrix} * \begin{bmatrix} Y_0 \\ \vdots \\ Y_N \end{bmatrix} = X_0Y_0 + X_1Y_1 \ldots X_NY_N$

Primary Stage

Cycle #	Primary Stage Computations			
	SP0	SP1	SP2	SP3
1	X_0Y_0 +	X_1Y_1 +	X_2Y_2 +	X_3Y_3
2	X_4Y_4 +	X_5Y_5 +	X_6Y_6 +	X_7Y_7
3	X_8Y_8 +	X_9Y_9 +	$X_{10}Y_{10}$ +	$X_{11}Y_{11}$
⋮		⋮		
N	$X_{N-3}Y_{N-3}$ +	$X_{N-2}Y_{N-2}$ +	$Y_{N-1}Y_{N-1}$ +	X_NY_N

Shadow Stage

Cycle #	Shadow Stage Computations			
	SP0	SP1	SP2	SP3
1	No operation			
2	No operation			
3	X_1Y_0 +	X_2Y_1 +	X_3Y_2 +	X_4Y_3
4	X_5Y_4 +	X_6Y_5 +	X_7Y_6 +	X_8Y_7
⋮		⋮		
N+2	$X_{N-2}Y_{N-3}$ +	$X_{N-1}Y_{N-2}$ +	X_NY_{N-1} +	$X_{N+1}Y_N$

$\begin{bmatrix} X_1 \\ \vdots \\ X_{N+1} \end{bmatrix} * \begin{bmatrix} Y_0 \\ \vdots \\ Y_N \end{bmatrix}$ (Shuffle X' Left by one)

Subsequent Cycles

Primary Stage Cycle #		Shadow Stage Cycle #	
N+1 ⋮ 2N	$\begin{bmatrix} X_2 \\ \vdots \\ X_{N+2} \end{bmatrix} * \begin{bmatrix} Y_0 \\ \vdots \\ Y_N \end{bmatrix}$	N+3 ⋮ N+5	$\begin{bmatrix} X_3 \\ \vdots \\ X_{N+3} \end{bmatrix} * \begin{bmatrix} Y_0 \\ \vdots \\ Y_N \end{bmatrix}$
N+4 ⋮ 3N	$\begin{bmatrix} X_4 \\ \vdots \\ X_{N+4} \end{bmatrix} * \begin{bmatrix} Y_0 \\ \vdots \\ Y_N \end{bmatrix}$	N+5 ⋮ N+7	$\begin{bmatrix} X_5 \\ \vdots \\ X_{N+5} \end{bmatrix} * \begin{bmatrix} Y_0 \\ \vdots \\ Y_N \end{bmatrix}$

⋮

FIG. 2B

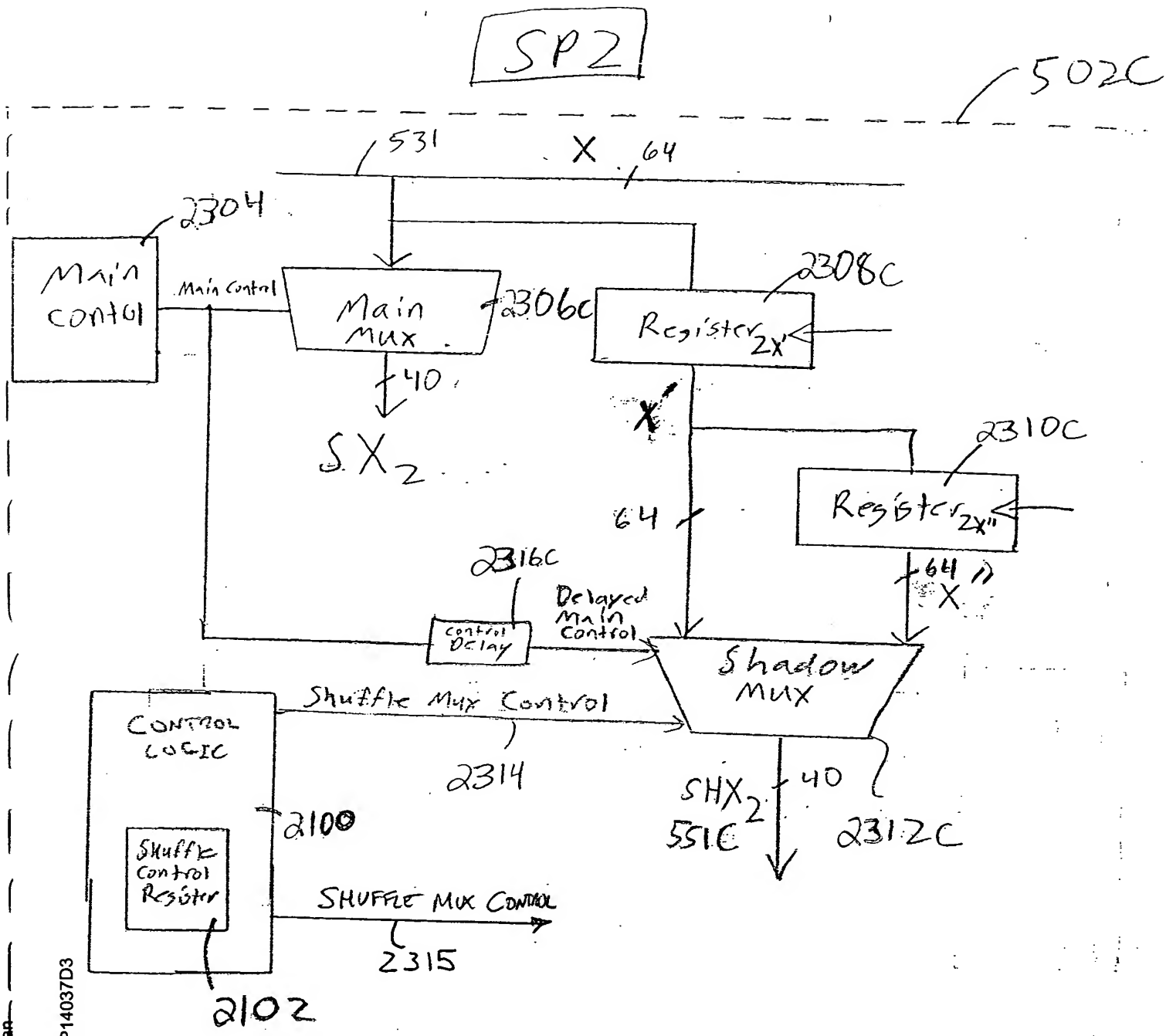


FIG. 23A

FIG. 23A FIG. 23B

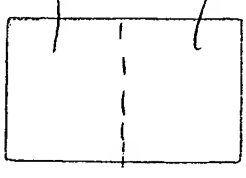


FIG 23

Blakely, Sokoloff, Taylor & Zafman LLP (714) 557-3800
Title: Memory with Memory Clusters for Power Reduction in an Integrated Circuit
1st Named Inventor: Ruban Kanapathippillai
Express Mail No.: EV323393502US Docket No.: 42P14037D3
Sheet: 33 of 58

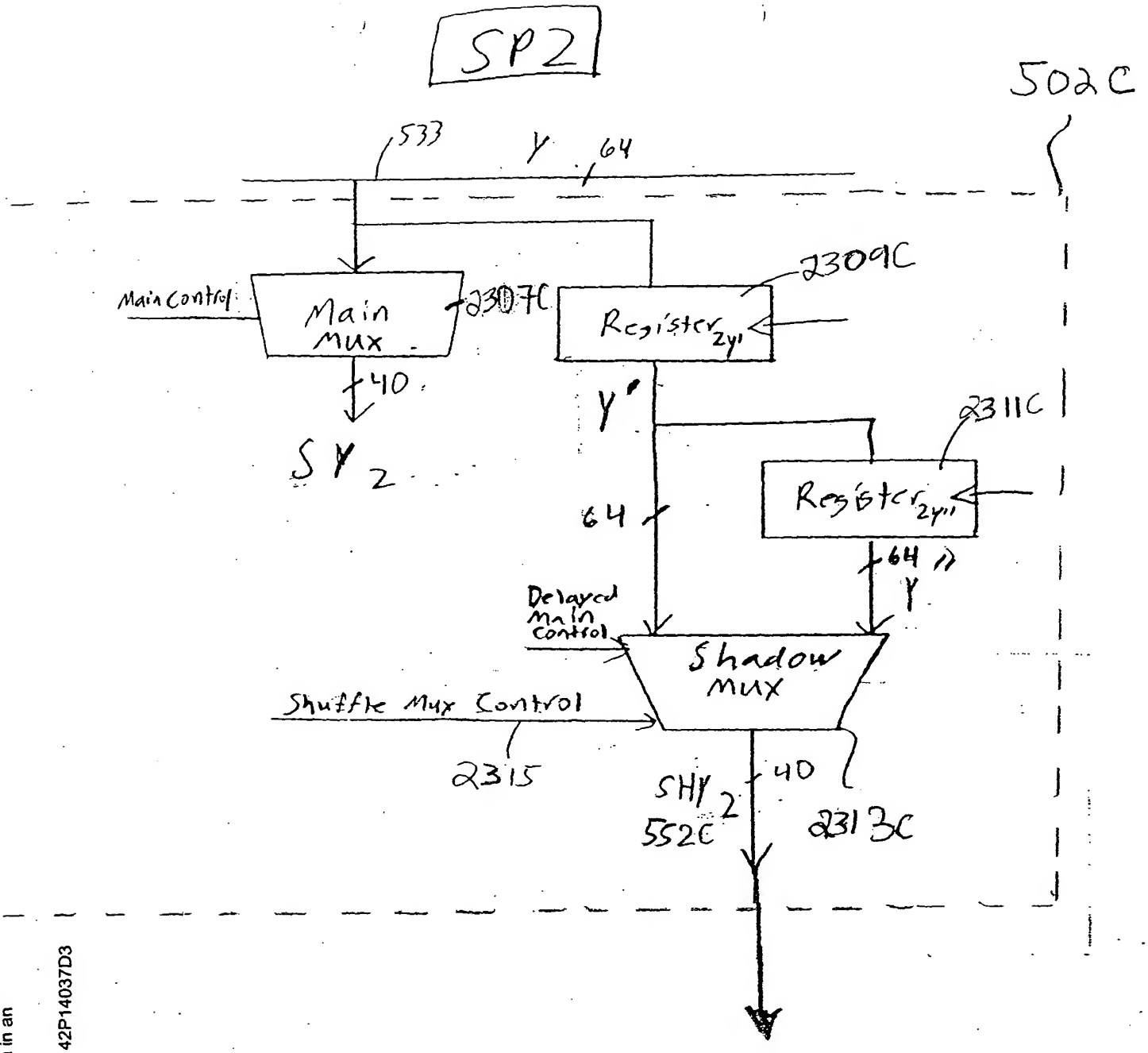
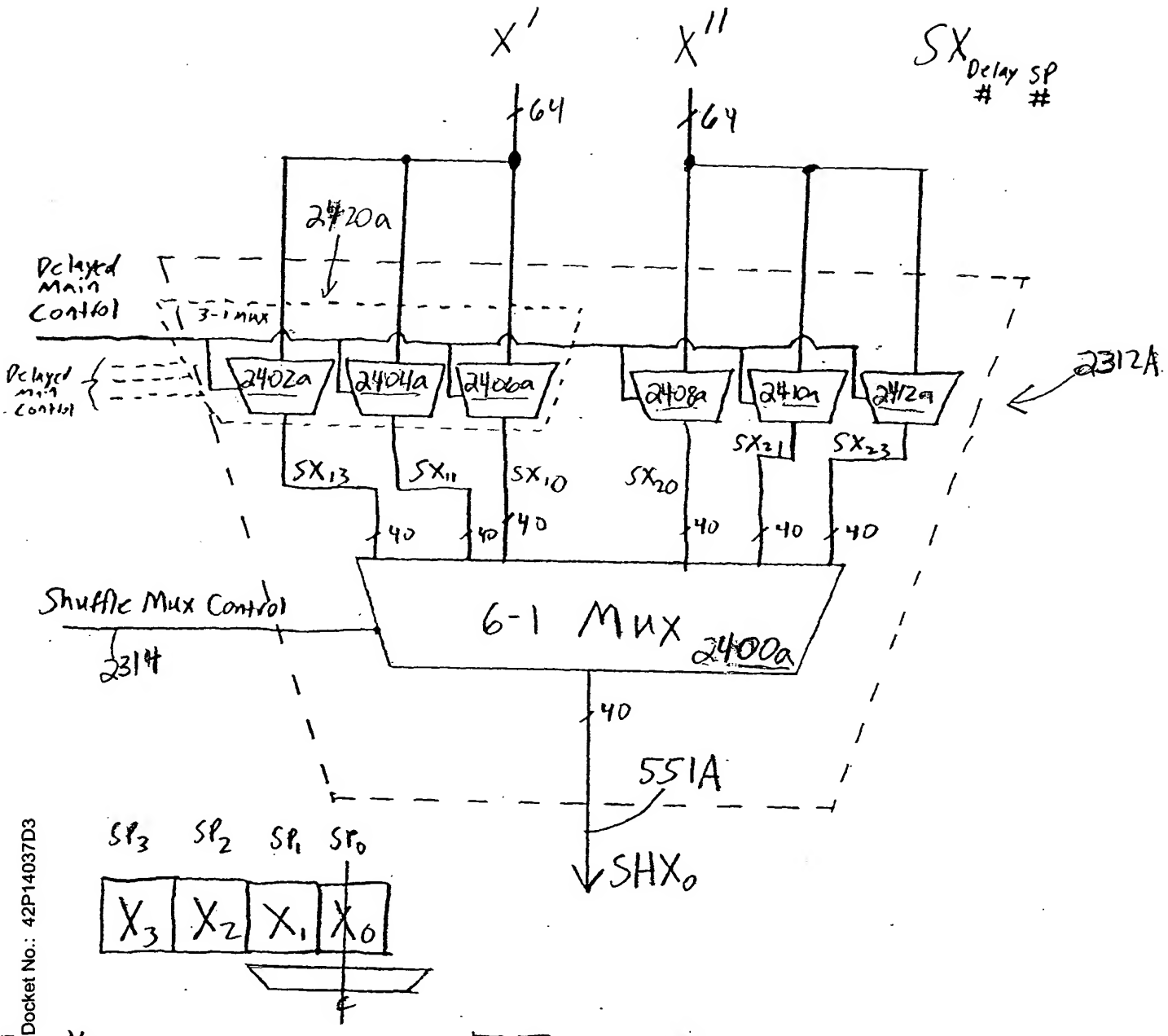


FIG. 23B

SPO Shadow Mux



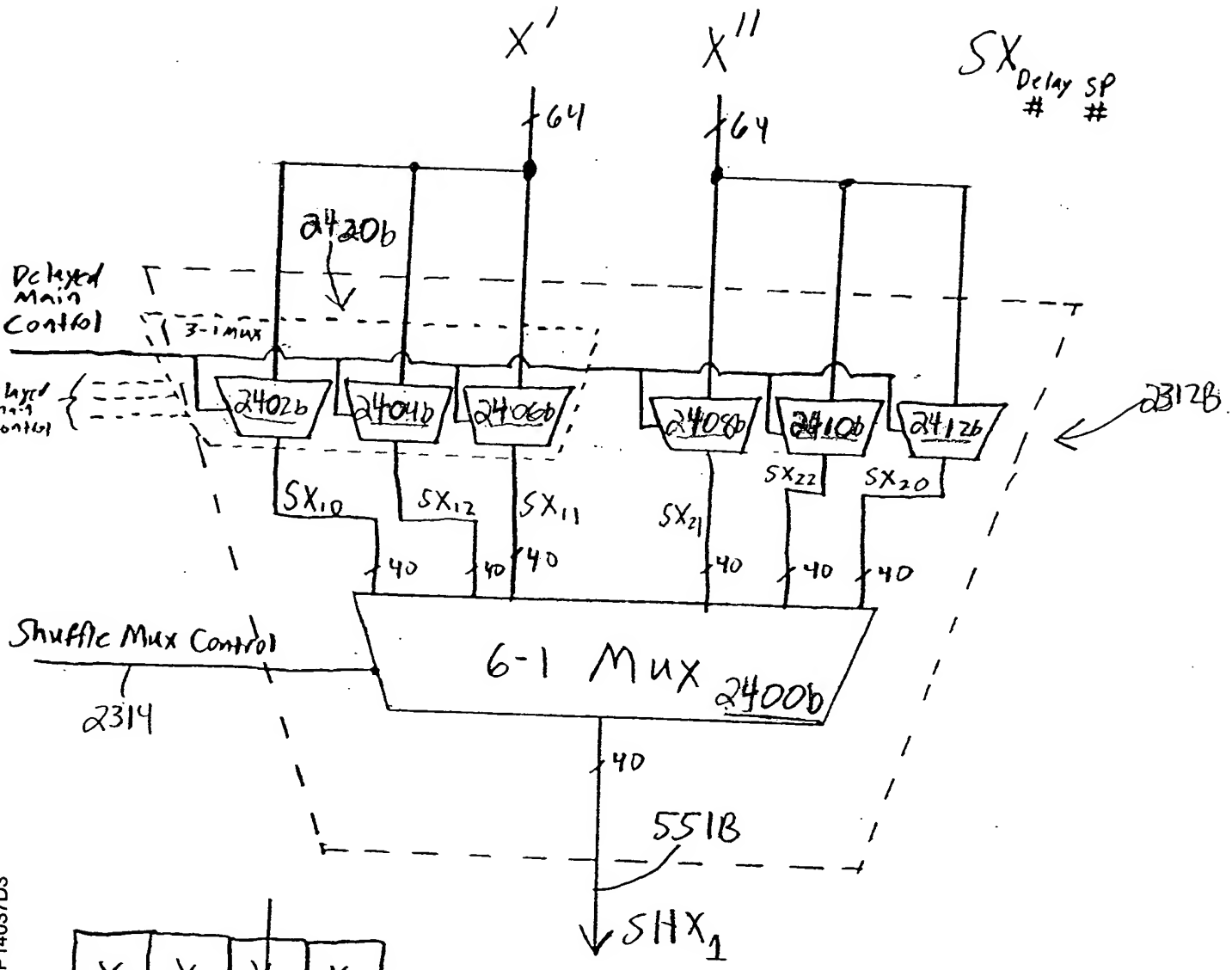
$X_0 = SX_{10}, SX_{20}$

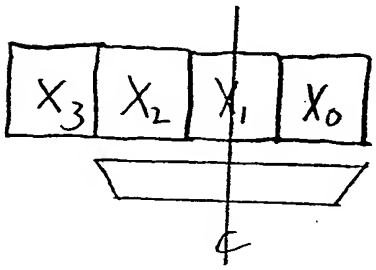
$X_1 = SX_{11}, SX_{21}$

$X_3 = SX_{13}, SX_{23}$

FIG. 24A

SP1
Shadow
Mux

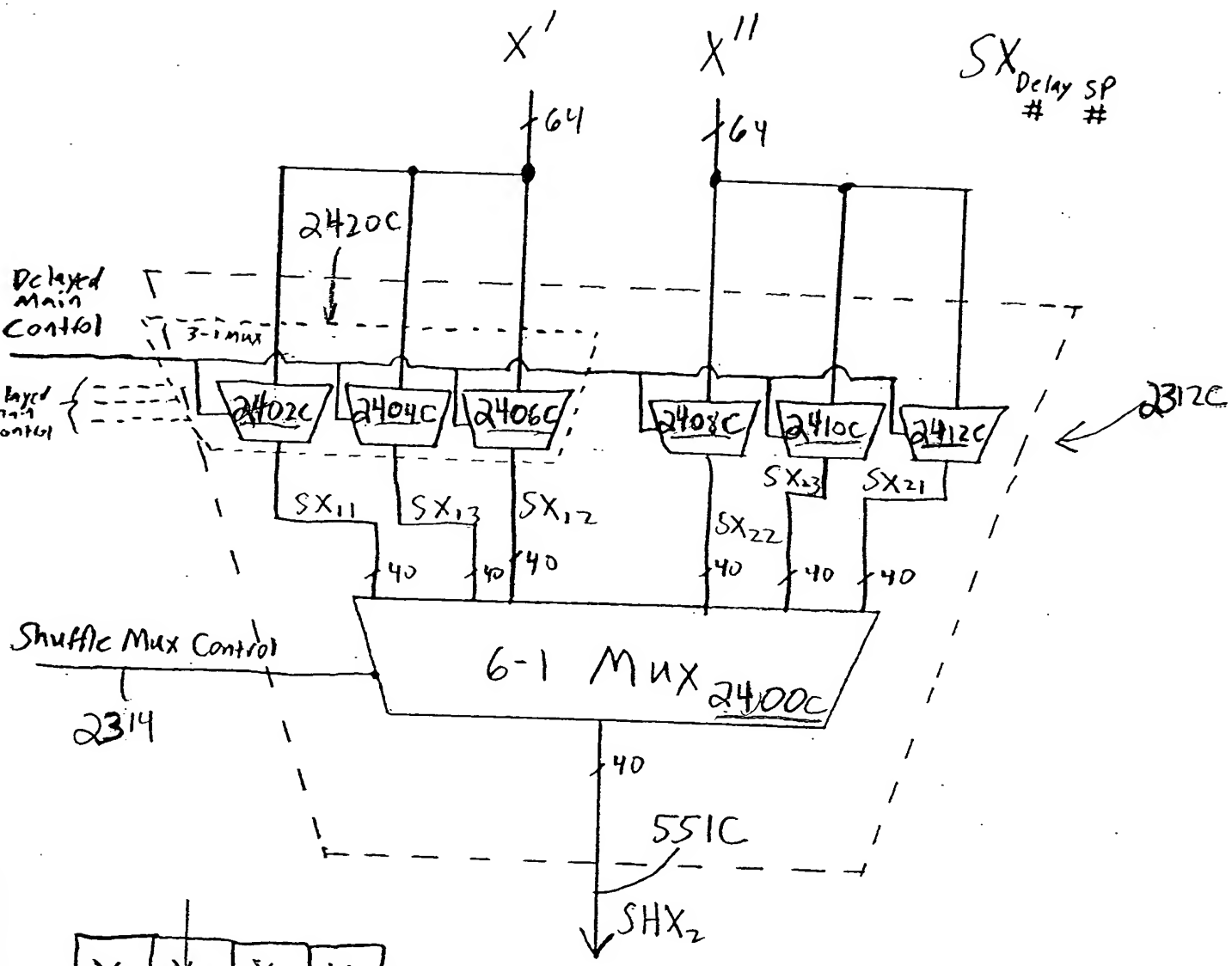


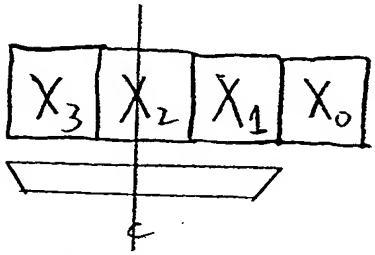


$X_1 = SX_{11}, SX_{21}$
$X_2 = SX_{12}, SX_{22}$
$X_0 = SX_{10}, SX_{20}$

FIG. 24B

SP2
Shadow
Mux

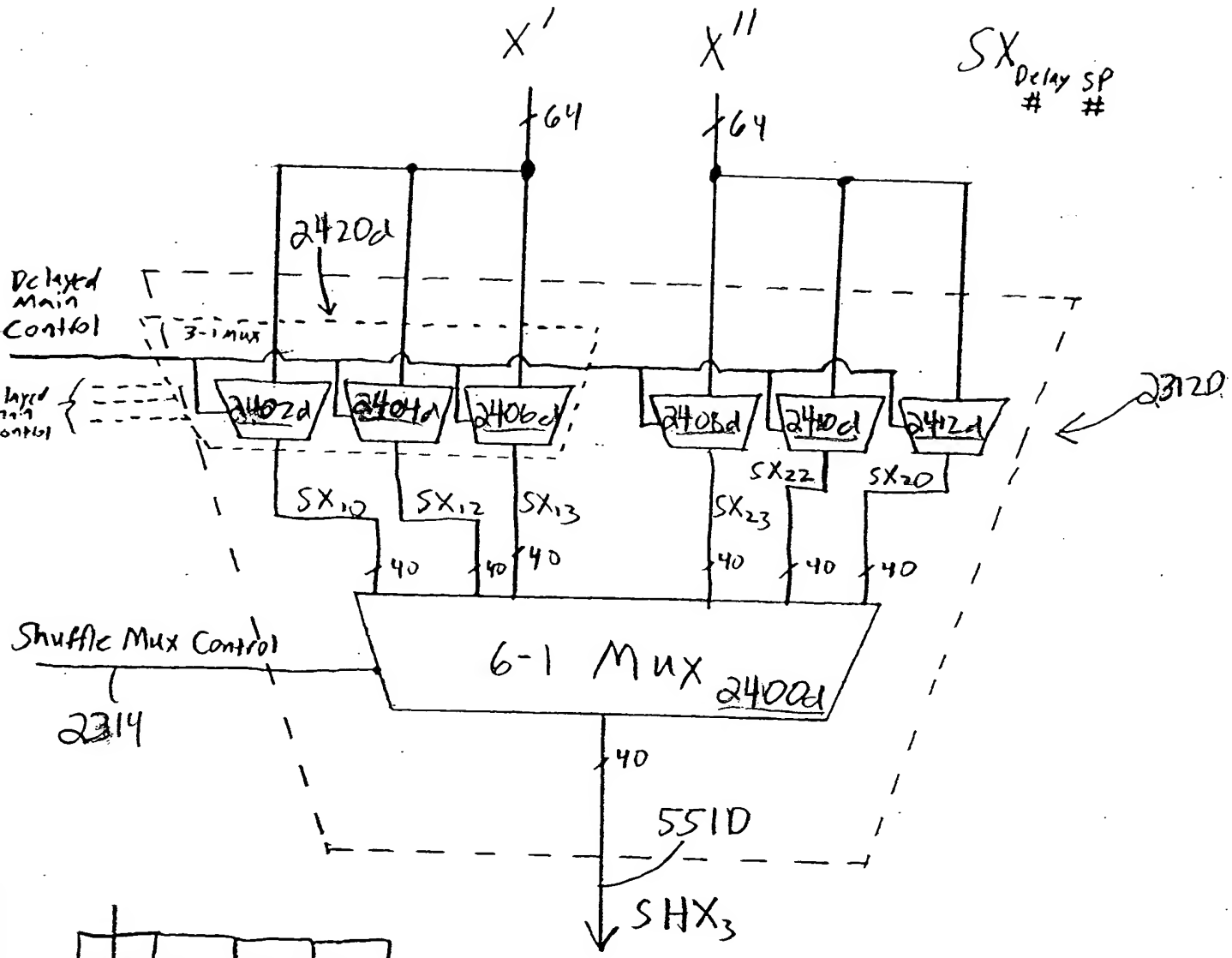


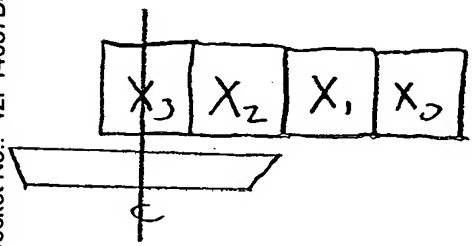


$X_2 = SX_{12}, SX_{22}$
$X_3 = SX_{13}, SX_{23}$
$X_1 = SX_{11}, SX_{21}$

FIG. 24C

SP3
Shadow
Mux





$X_3 = SX_{13}, SX_{23}$
$X_0 = SX_{10}, SX_{20}$
$X_2 = SX_{12}, SX_{22}$

FIG. 24D

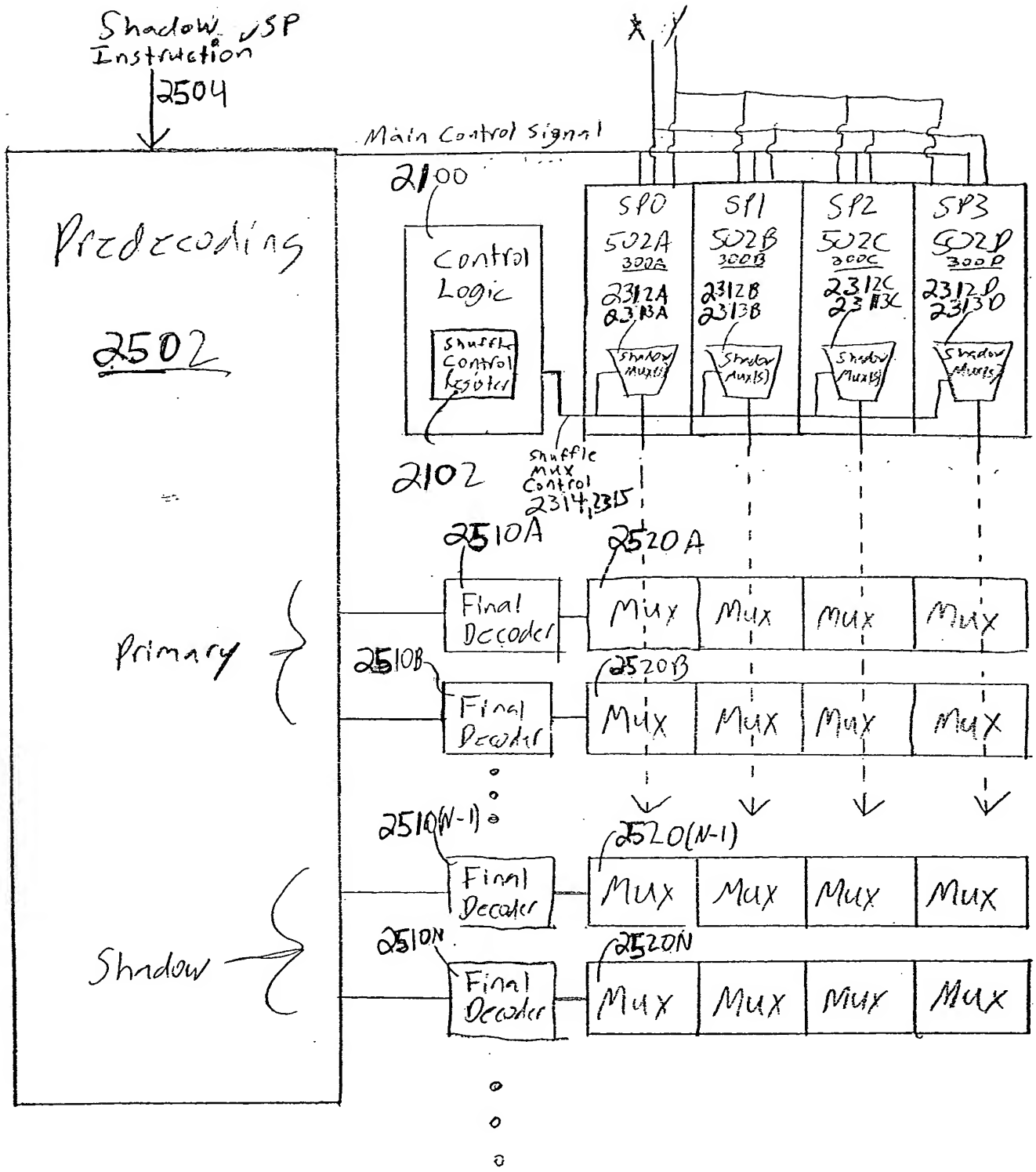


FIG. 25

Blakely, Sokoloff, Taylor & Zafman LLP (714) 557-3800
Title: Memory with Memory Clusters for Power Reduction in an Integrated Circuit
1st Named Inventor: Ruban Kanapathippillai
Express Mail No.: EV323393502US Docket No.: 42P14037D3
Sheet: 39 of 58

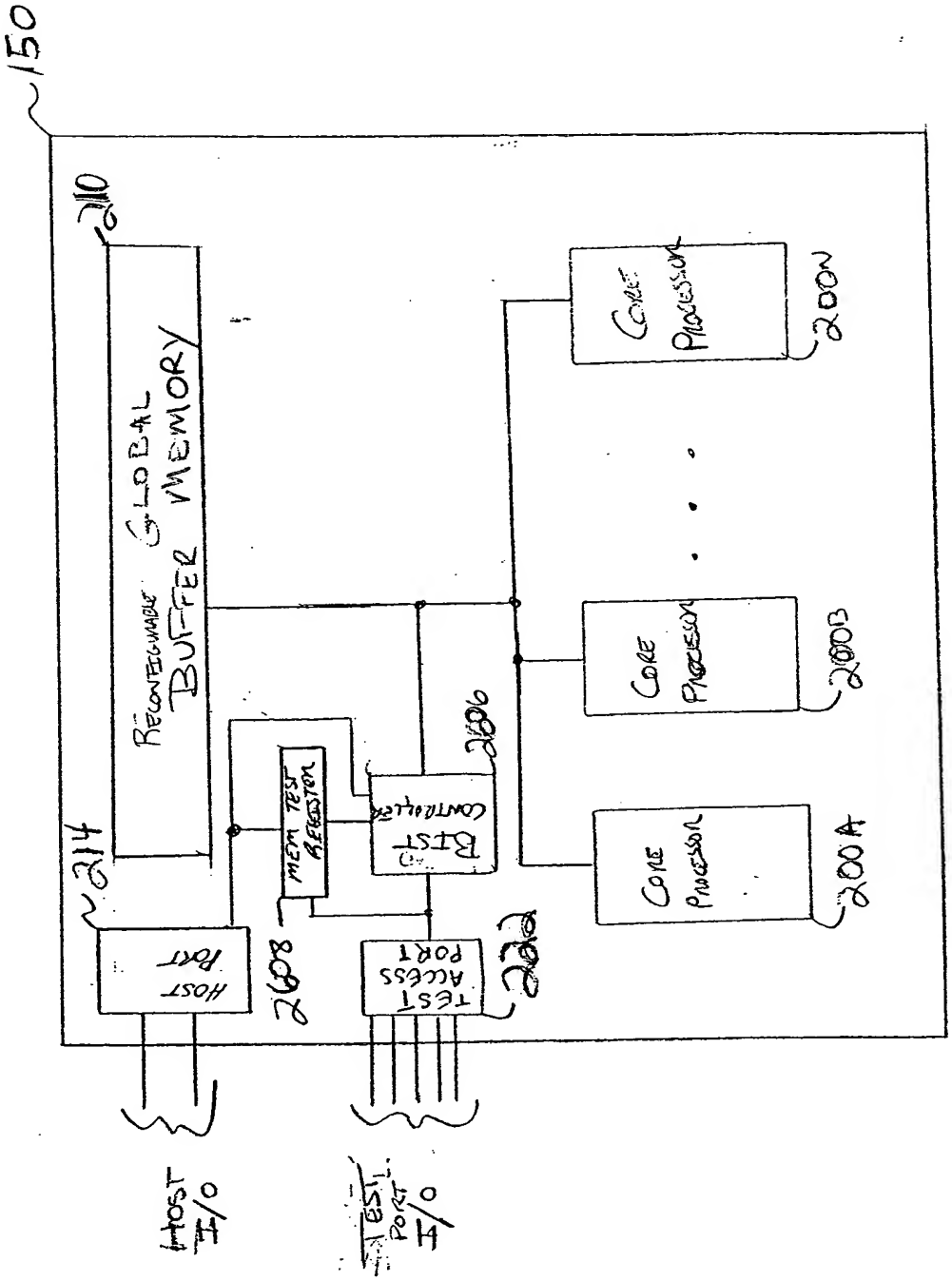

150
210
RECONFIGURABLE GLOBAL BUFFER MEMORY
214
HOST PORT
208
MEM TEST ARBITER
206
BIST CONTROLLER
TEST ACCESS PORT
204
CORE PROCESSOR
200N
CORE PROCESSOR
200B
CORE PROCESSOR
200A
HOST I/O
TEST PORT I/O

FIG. 2G

210

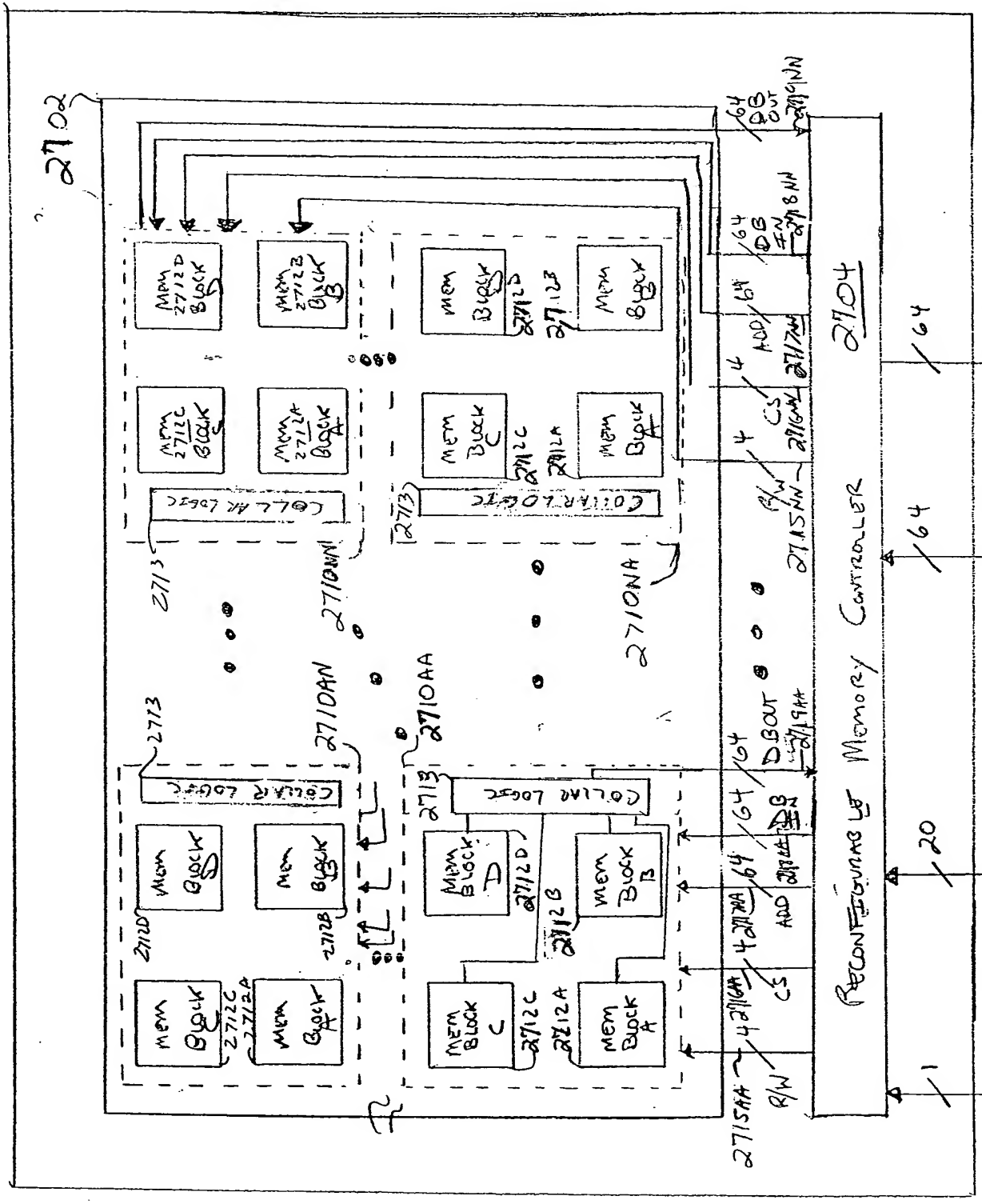

2702
2704
2705
2706
2707
2708
2709
2710AA
2710AN
2710NA
2710NN
2711A
2711B
2711C
2711D
2712A
2712B
2712C
2712D
2713
MEMORY CONTROLLER
DATA OUT
DATA IN
ADDRESS
R/W
CS
DBOUT
DBIN
ADD

FIG. 27

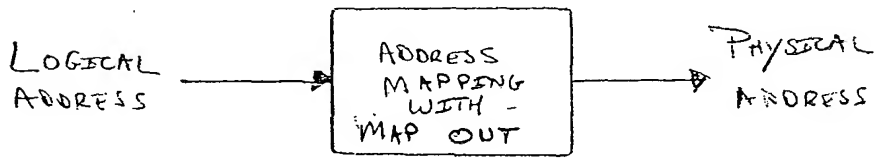

LOGICAL ADDRESS
ADDRESS MAPPING WITH - MAP OUT
PHYSICAL ADDRESS

FIG. 28

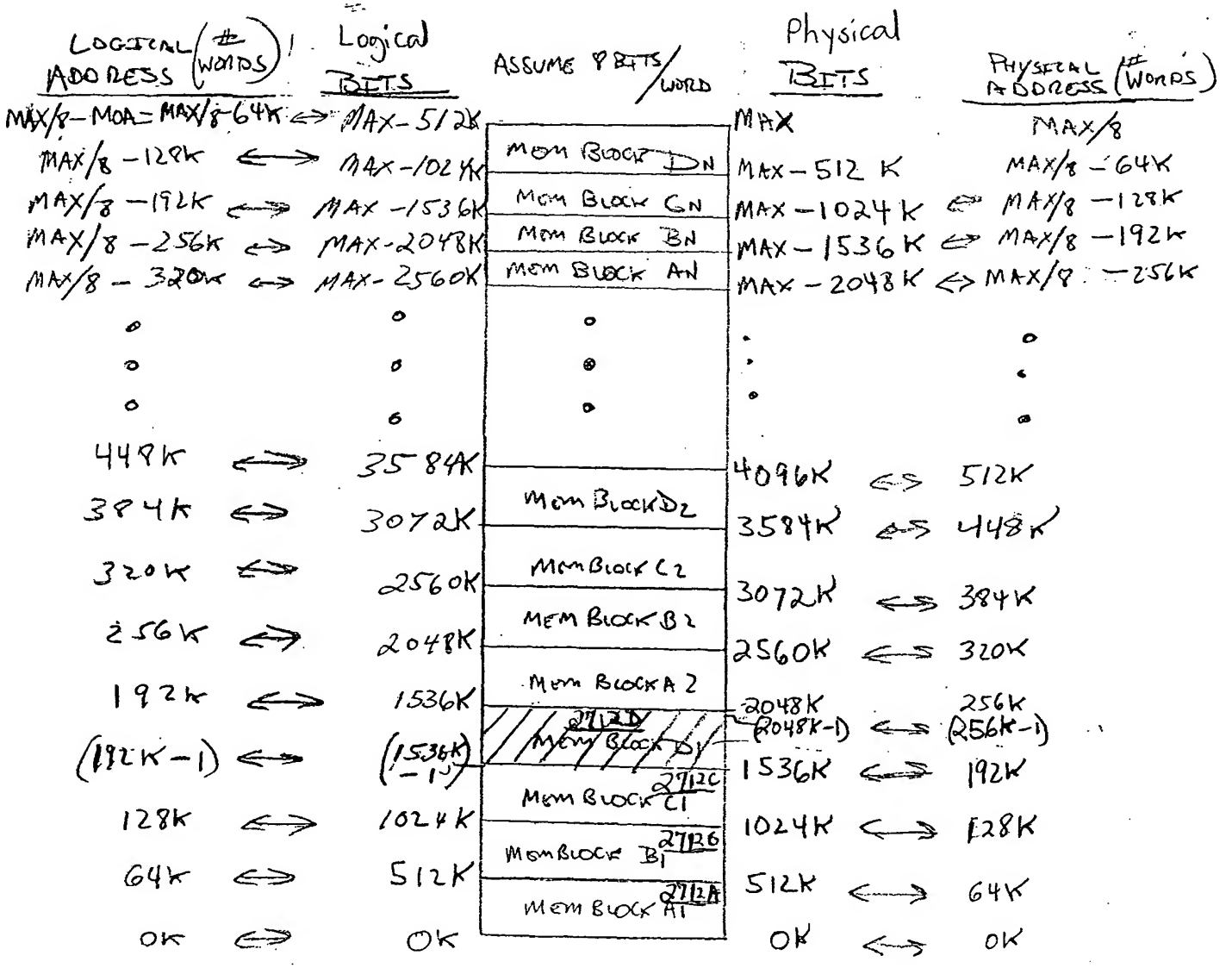

LOGICAL ADDRESS (# WORDS)
Logical BITS
ASSUME 8 BITS/WORD
Physical BITS
PHYSICAL ADDRESS (# WORDS)
MAX/8 - MOA = MAX/8 - 64K ⟷ MAX - 512K
MAX/8 - 128K ⟷ MAX - 1024K
MAX/8 - 192K ⟷ MAX - 1536K
MAX/8 - 256K ⟷ MAX - 2048K
MAX/8 - 320K ⟷ MAX - 2560K
MAX
MAX - 512 K
MAX - 1024 K
MAX - 1536 K
MAX - 2048 K
MAX/8
MAX/8 - 64K
MAX/8 - 128K
MAX/8 - 192K
MAX/8 - 256K
MEM BLOCK DN
MEM BLOCK GN
MEM BLOCK BN
MEM BLOCK AN
448K ⟷ 3584K
384K ⟷ 3072K
320K ⟷ 2560K
256K ⟷ 2048K
192K ⟷ 1536K
(192K-1) ⟷ (1536K-1)
128K ⟷ 1024K
64K ⟷ 512K
0K ⟷ 0K
4096K ⟷ 512K
3584K ⟷ 448K
3072K ⟷ 384K
2560K ⟷ 320K
2048K ⟷ 256K
(2048K-1) ⟷ (256K-1)
1536K ⟷ 192K
1024K ⟷ 128K
512K ⟷ 64K
0K ⟷ 0K
MEM BLOCK D2
MEM BLOCK C2
MEM BLOCK B2
MEM BLOCK A2
MEM BLOCK D1
MEM BLOCK C1
MEM BLOCK B1
MEM BLOCK A1
2712D
2712C
2712B
2712A

FIG. 29

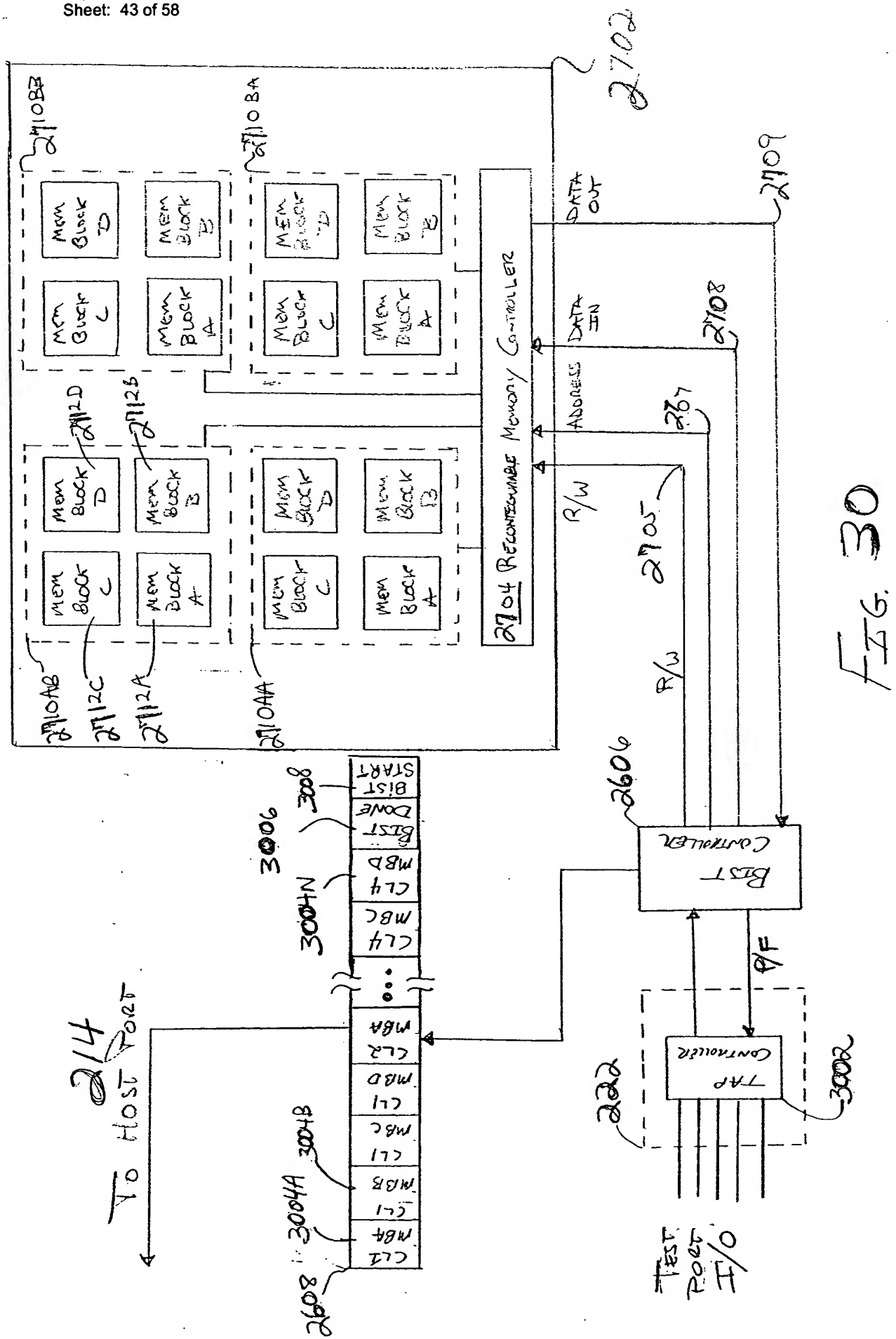

2702
2710BB
2710BA
MEM BLOCK D
MEM BLOCK B
MEM BLOCK C
MEM BLOCK A
2704 RECONFIGURABLE MEMORY CONTROLLER
DATA OUT
2709
DATA IN
2708
ADDRESS
2707
R/W
2705
2712D
2712B
2712C
2712A
2710AB
2710AA
R/W
2606
BIST CONTROLLER
3006
3008
BIST START
BIST POWER
3004N
CLK MBC
CLK MBA
3004B
CLK MBB
CLK MBA
3004A
2608
2602
TAP CONTROLLER
3002
T/F
2202
TEST I/O PORT
2214
TO HOST PORT

FIG. 30

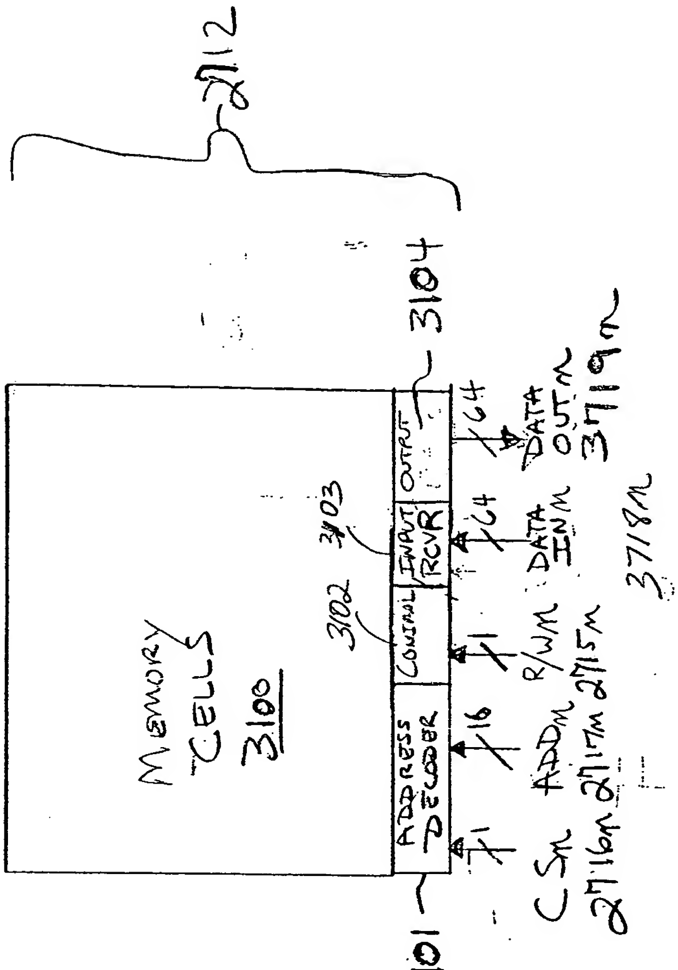

2712
MEMORY CELLS
3100
ADDRESS DECODER
CONTROL
INPUT RCVR
OUTPT
3101
3102
3103
3104
1
16
1
64
64
CS
ADDR
R/W
DATA IN
DATA OUT

FIG. 31

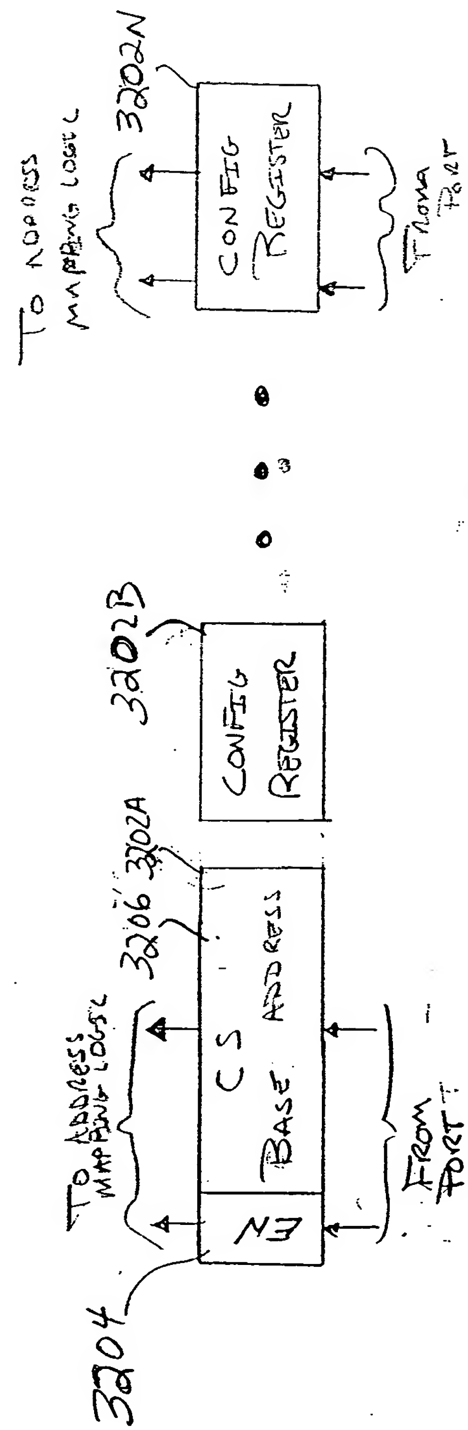

3202N
CONFIG REGISTER
3202B
CONFIG REGISTER
3202A
3206
CS BASE ADDRESS
EN
3204
TO ADDRESS MAPPING LOGIC
FROM PORT

FIG. 32

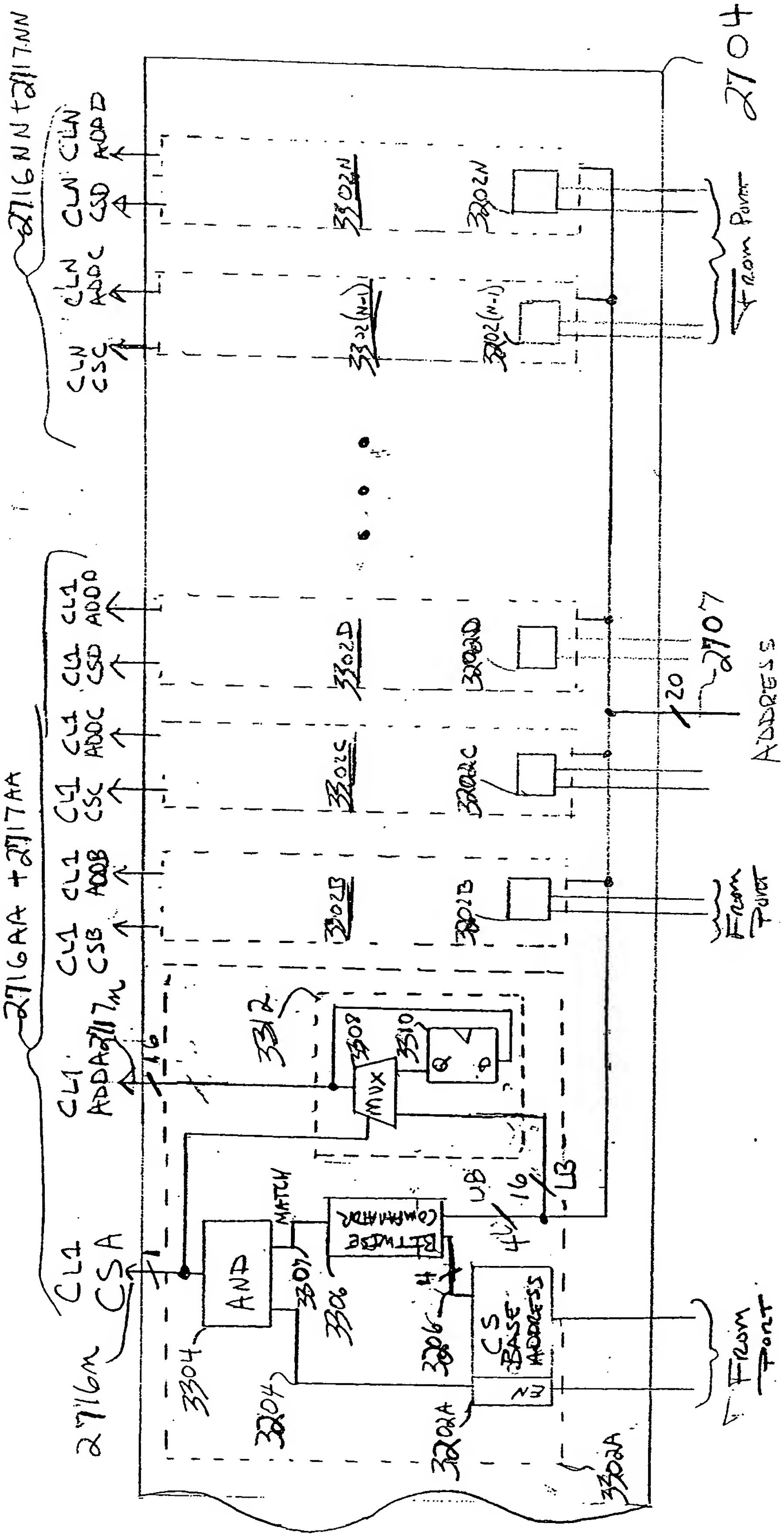

AND
MUX
BITWISE COMPARATOR
MATCH
CS BASE ADDRESS
EN
UB
LB
From Port
ADDRESS
2704
2707
3304
3306
3308
3310
3312

FIG. 33A

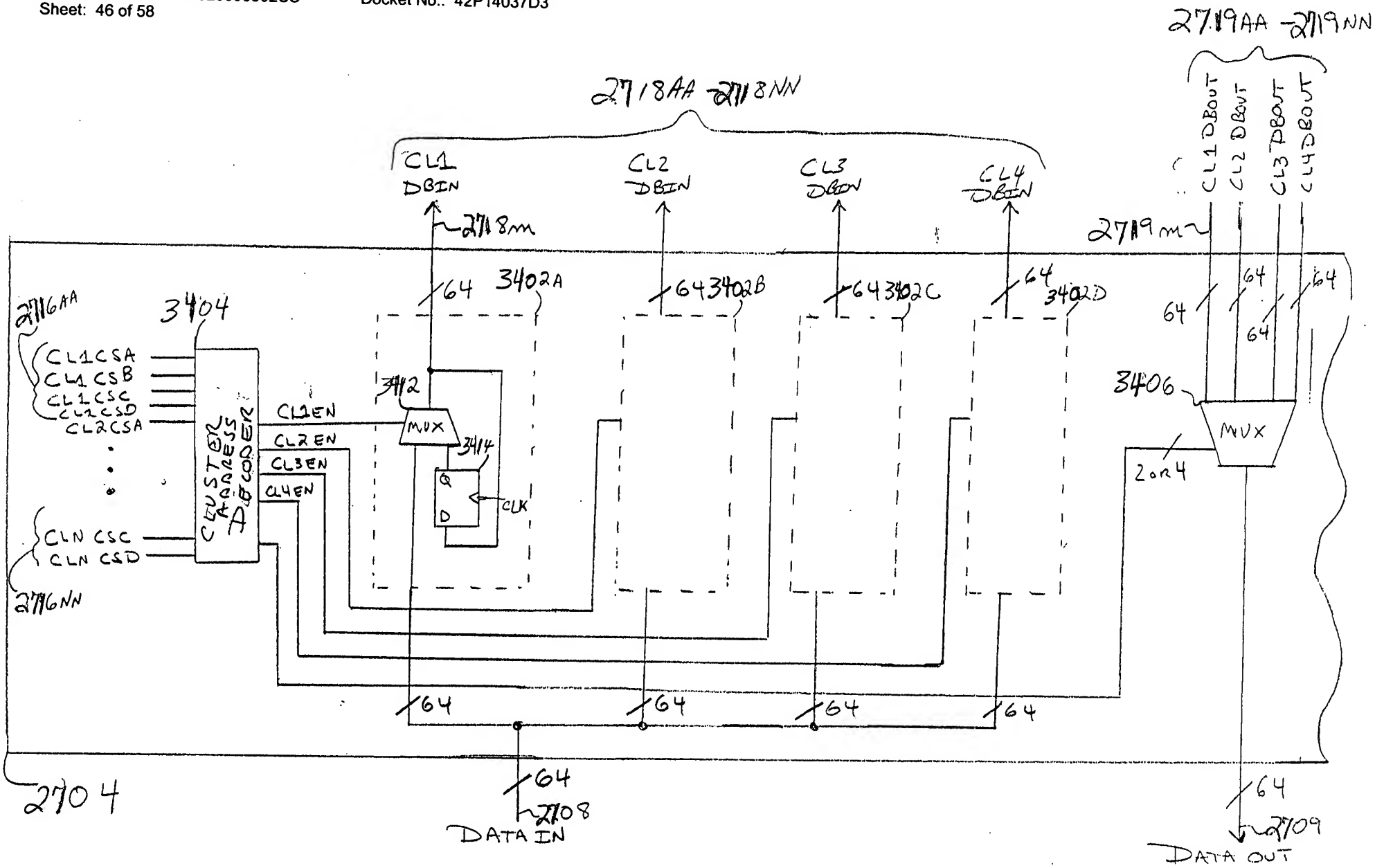

2719AA - 2719NN
2718AA - 2718NN
CL1 DBIN
CL2 DBIN
CL3 DBIN
CL4 DBIN
2718m
2719m
CL1DBOUT
CL2DBOUT
CL3DBOUT
CL4DBOUT
64
3402A
3402B
3402C
3402D
2716AA
3404
CL1CSA
CL1CSB
CL1CSC
CL2CSD
CL2CSA
CLN CSC
CLN CSD
2716NN
CLUSTER ADDRESS DECODER
CL1EN
CL2EN
CL3EN
CL4EN
3412
MUX
3414
CLK
Q
D
3406
MUX
2 or 4
2704
64
2708
DATA IN
2709
DATA OUT

FIG. 33B

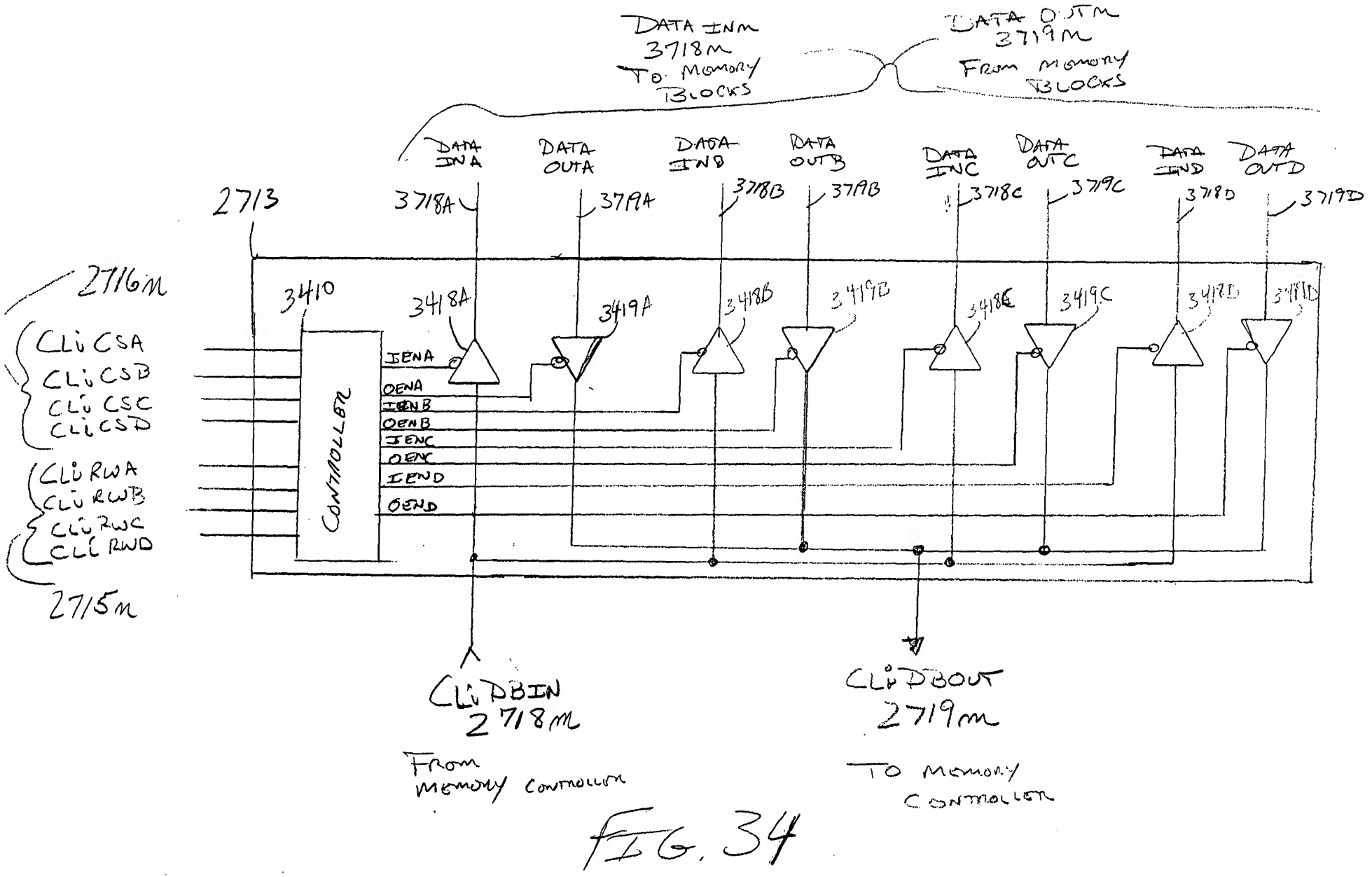

DATA INM
3718M
TO MEMORY BLOCKS
DATA OUTM
3719M
FROM MEMORY BLOCKS
DATA INA
DATA OUTA
DATA INB
DATA OUTB
DATA INC
DATA OUTC
DATA IND
DATA OUTD
2713
3718A
3719A
3718B
3719B
3718C
3719C
3718D
3719D
2716M
3410
3418A
3419A
3418B
3419B
3418C
3419C
3418D
3419D
CLUCSA
CLUCSB
CLUCSC
CLUCSD
CLURWA
CLURWB
CLURWC
CLURWD
2715M
CONTROLLER
IENA
OENA
IENB
OENB
IENC
OENC
IEND
OEND
CLUDBIN
2718M
FROM MEMORY CONTROLLER
CLUDBOUT
2719M
TO MEMORY CONTROLLER

FIG. 34

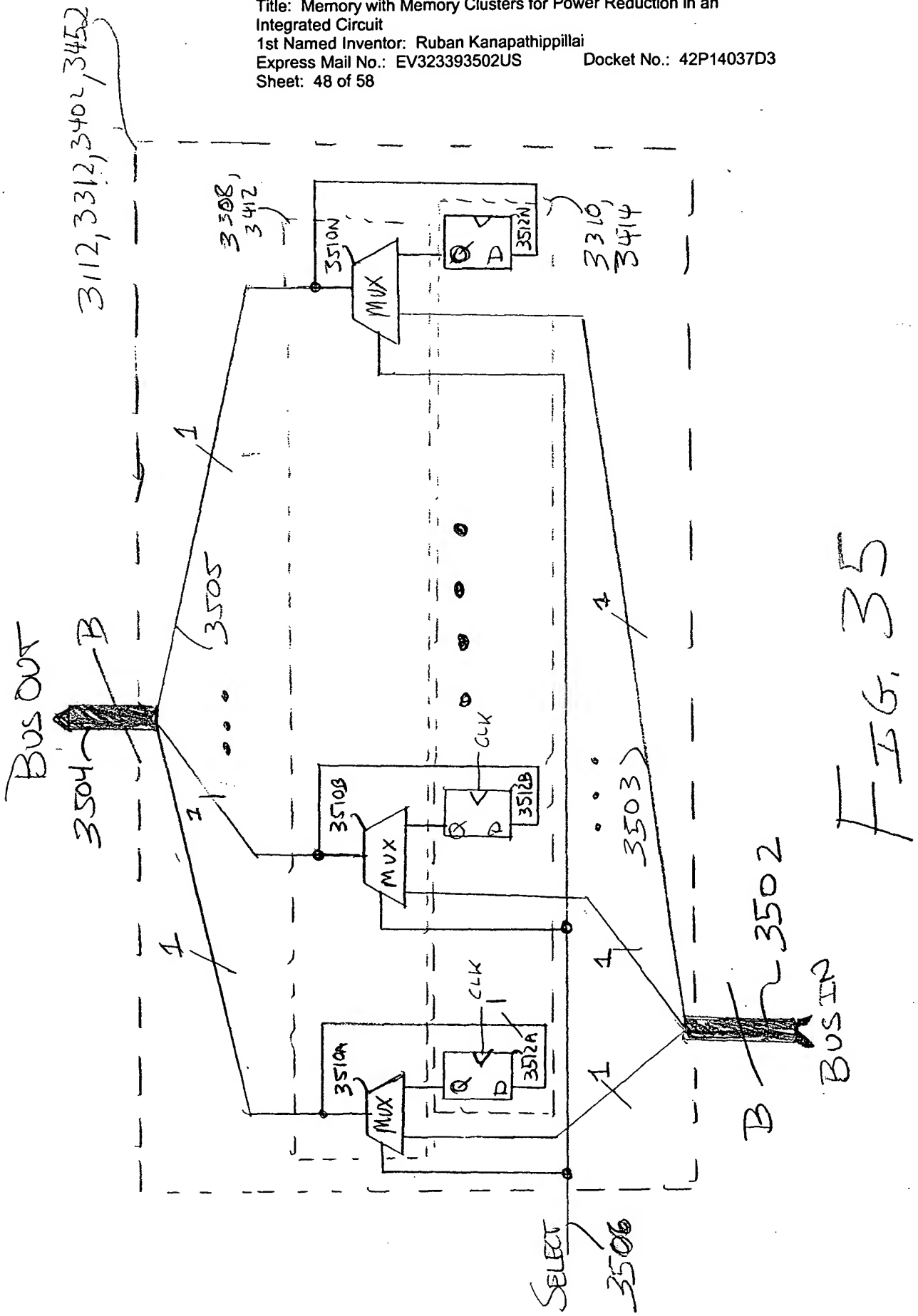

3112, 3312, 3402, 3452
3308, 3412
3310, 3414
3510N
MUX
3512N
3505
1
BUS OUT
3504
B
3510B
3512B
CLK
3503
3510A
3512A
3502
BUS IN
SELECT
3506

FIG. 35

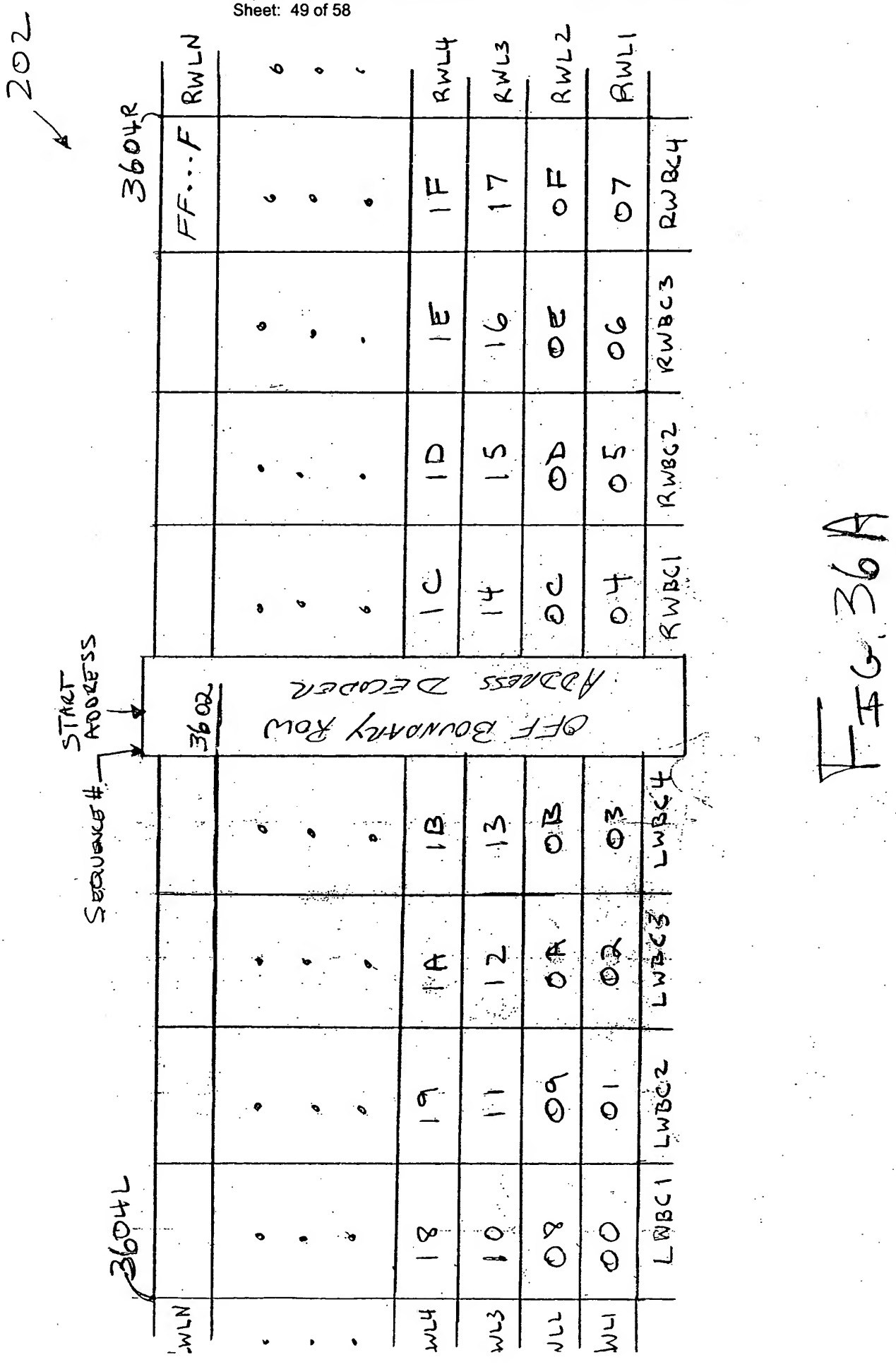

202
3604R
3604L
RWLN
RWL4
RWL3
RWL2
RWL1
FF...F
RWBC4
RWBC3
RWBC2
RWBC1
1F
17
0F
07
1E
16
0E
06
1D
15
0D
05
1C
14
0C
04
3602
OFF Boundary Row
Address Decoder
START ADDRESS
SEQUENCE #
LWBC4
LWBC3
LWBC2
LWBC1
1B
13
0B
03
1A
12
0A
02
19
11
09
01
18
10
08
00
WLN
WL4
WL3
WL2
WL1

FIG. 36A

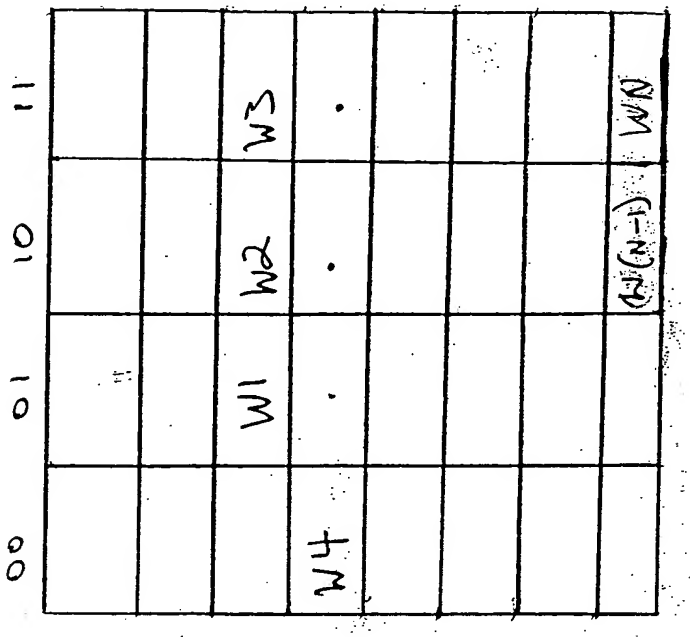

11
10
01
00
W3
W2
W1
W4
W(N-1)
WN
HARDWARE ASSIGNED VIEW
OFFSET PHYSICAL ADDRESS SPACE

FIG. 3C

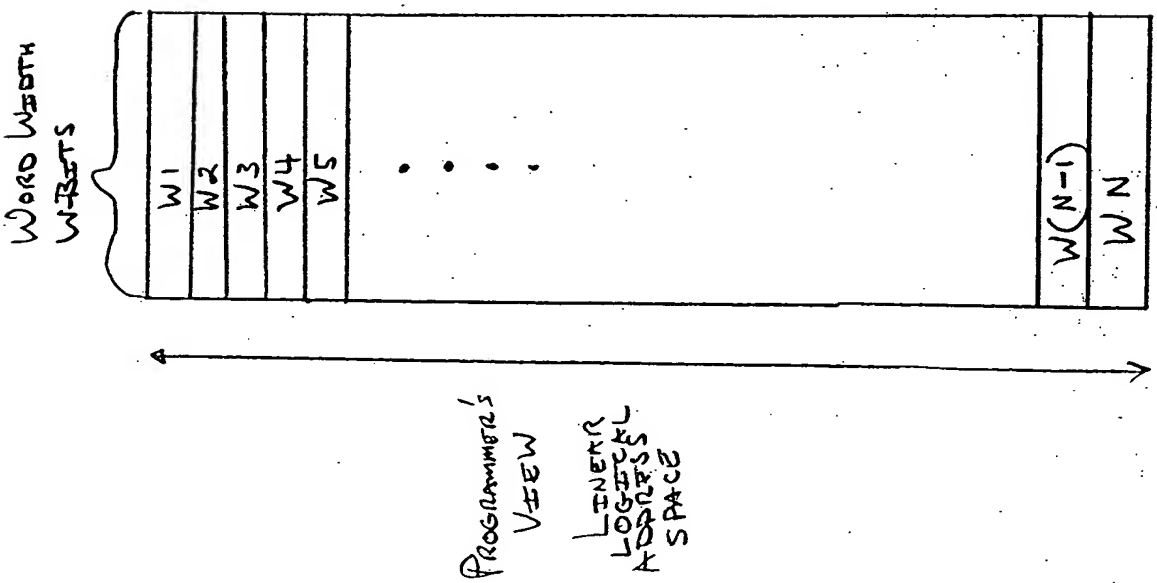

WORD WIDTH WIDTH
W1
W2
W3
W4
W5
W(N-1)
WN
PROGRAMMER'S VIEW
LINEAR LOGICAL ADDRESS SPACE

FIG. 3B

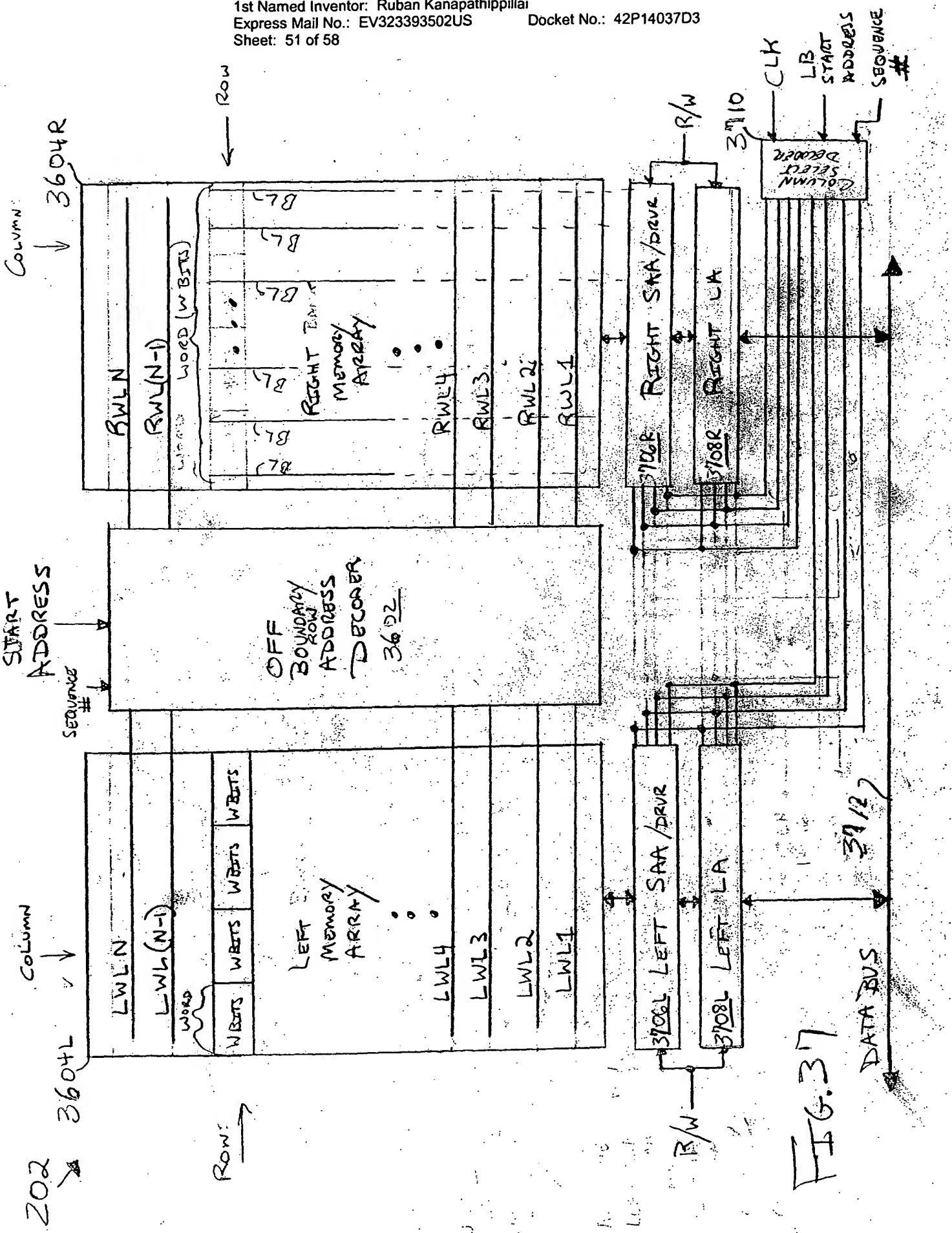

202
360L
360R
COLUMN
ROW
START ADDRESS
SEQUENCE #
OFF BOUNDARY ROW ADDRESS DECODER 362
LEFT MEMORY ARRAY
RIGHT MEMORY ARRAY
LWL1
LWL2
LWL3
LWL4
LWL(N-1)
LWLN
RWL1
RWL2
RWL3
RWL4
RWL(N-1)
RWLN
WORD (W BITS)
W BITS
370L LEFT SA/DRVR
370L LEFT LA
370R RIGHT SA/DRVR
370R RIGHT LA
R/W
CLK
LB START ADDRESS
SEQUENCE #
COLUMN SELECT DECODER
372
DATA BUS

FIG. 37

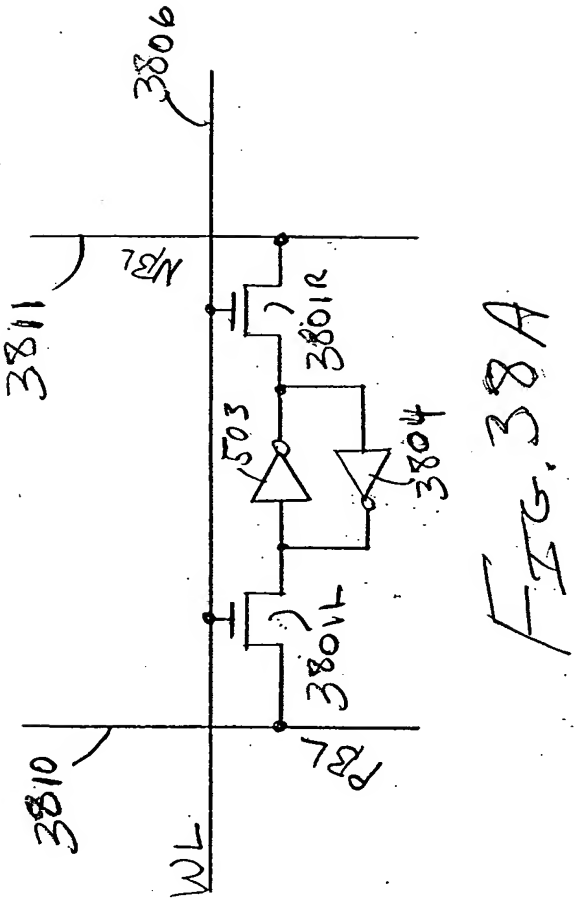


FIG. 38A

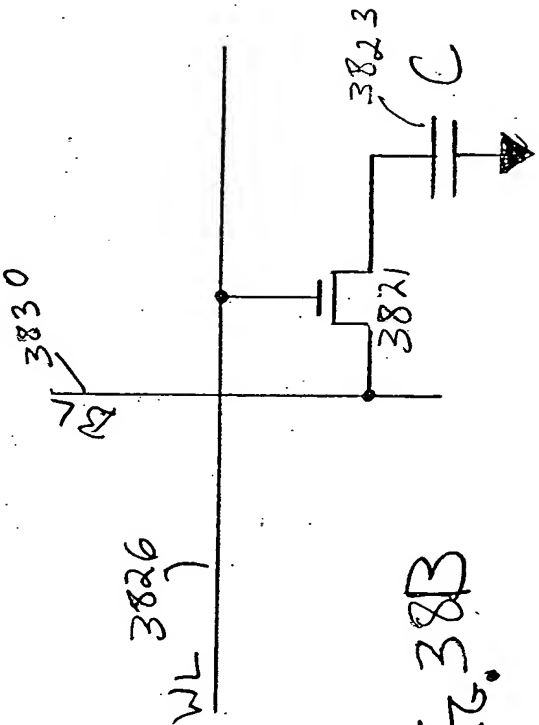


FIG. 38B

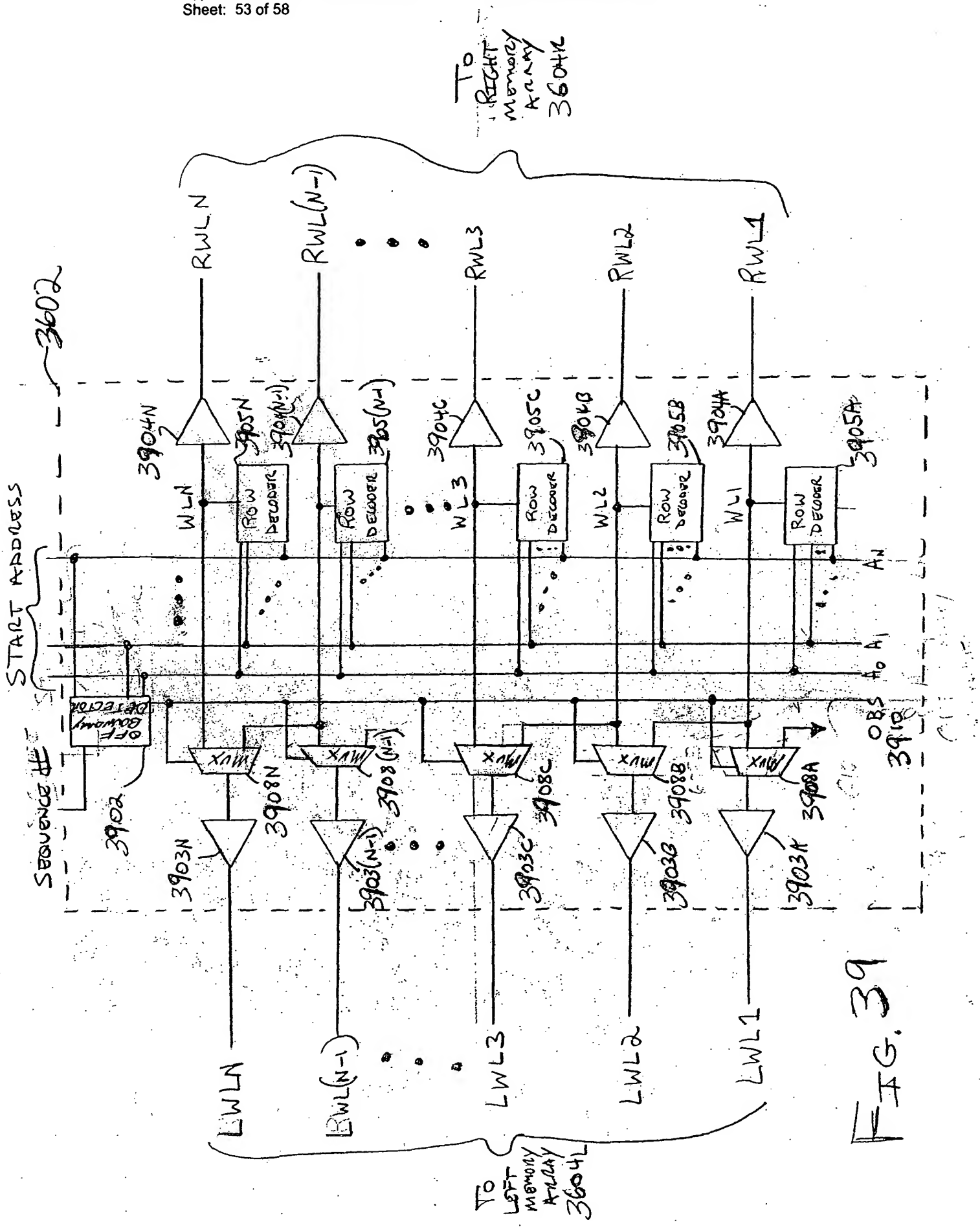

To Right Memory Array 3604R
RWLN
RWL(N-1)
RWL3
RWL2
RWL1
3602
3904N
3905N
3904(N-1)
3905(N-1)
3904C
3905C
3904B
3905B
3904A
3905A
WLN
WL3
WL2
WL1
ROW DECODER
START ADDRESS
SEQUENCE #
OFF Boundary Detector
3902
3908N
3908(N-1)
3908C
3908B
3908A
OBS 3910
A0
A1
AN
3903N
3903(N-1)
3903C
3903B
3903A
LWLN
LWL(N-1)
LWL3
LWL2
LWL1
To Left Memory Array 3604L

FIG. 39

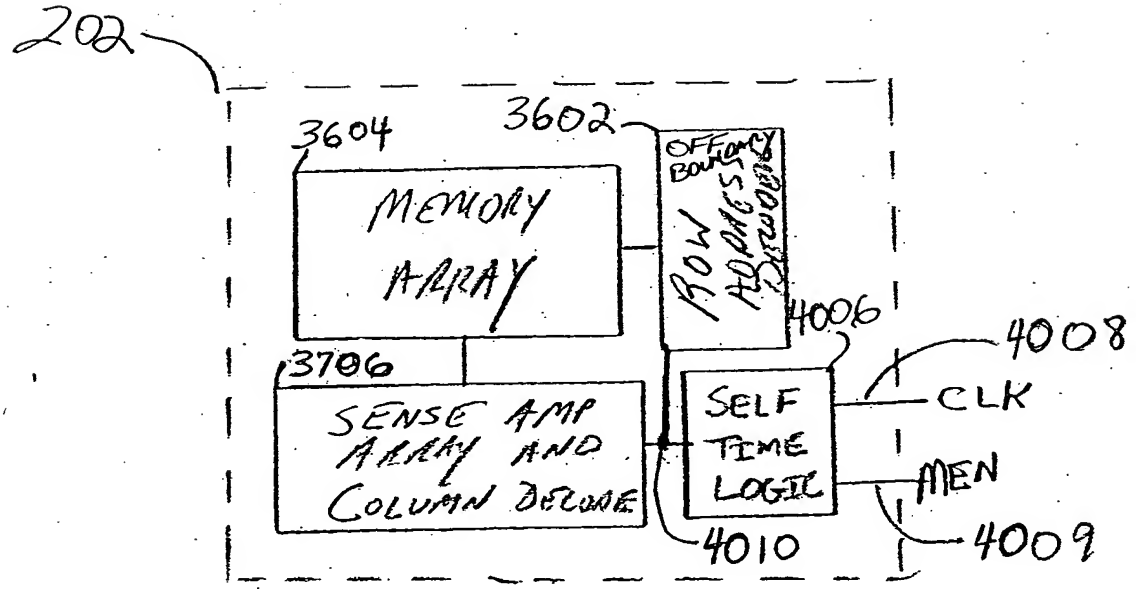

202
3604
3602
MEMORY ARRAY
OFFSET BOUNDARY
ROW ADDRESS DECODER
4006
4008
3706
SENSE AMP ARRAY AND COLUMN DECODE
SELF TIME LOGIC
CLK
MEN
4010
4009

FIG. 40

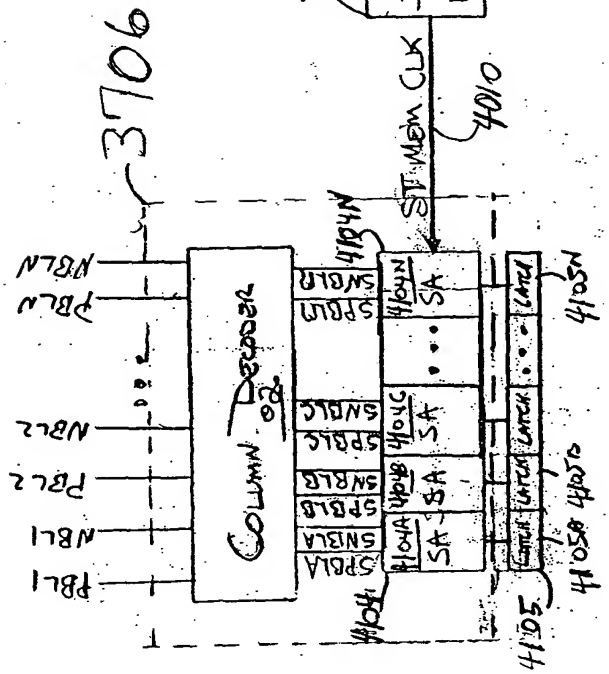


FIG. 41

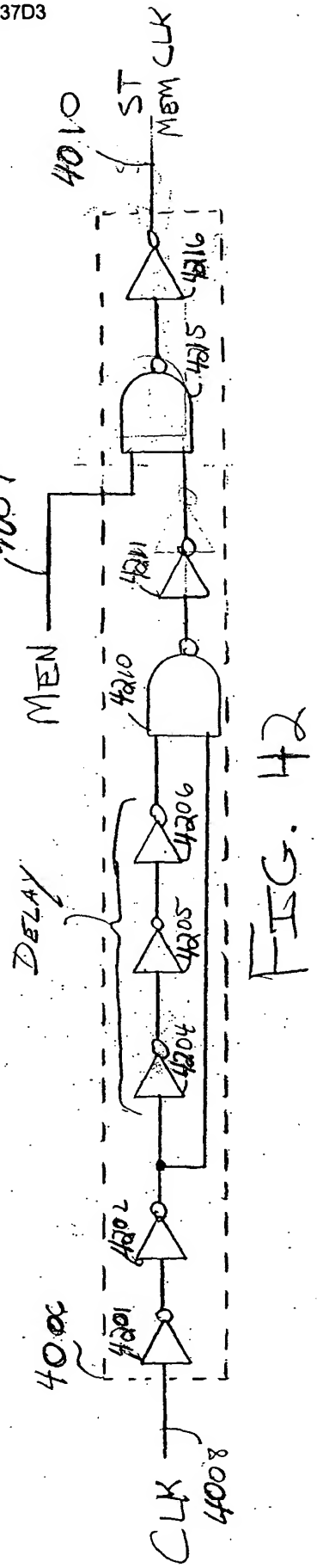


FIG. 42

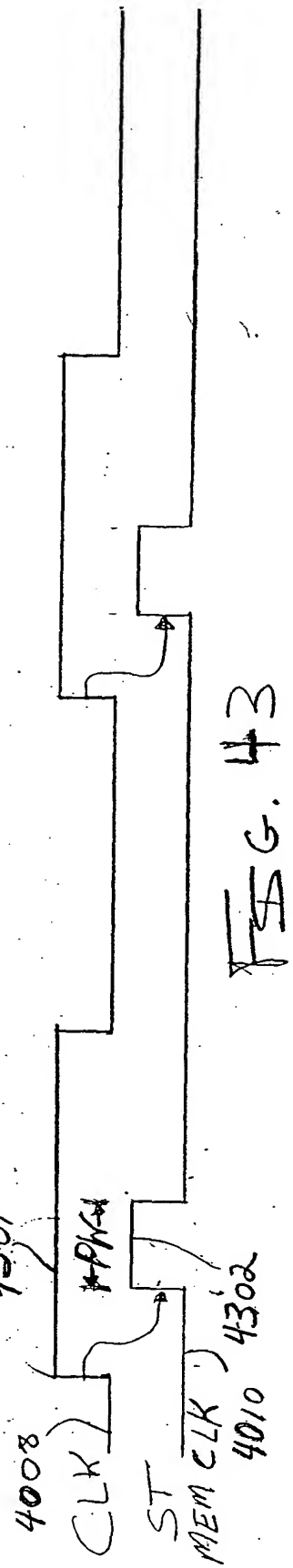


FIG. 43

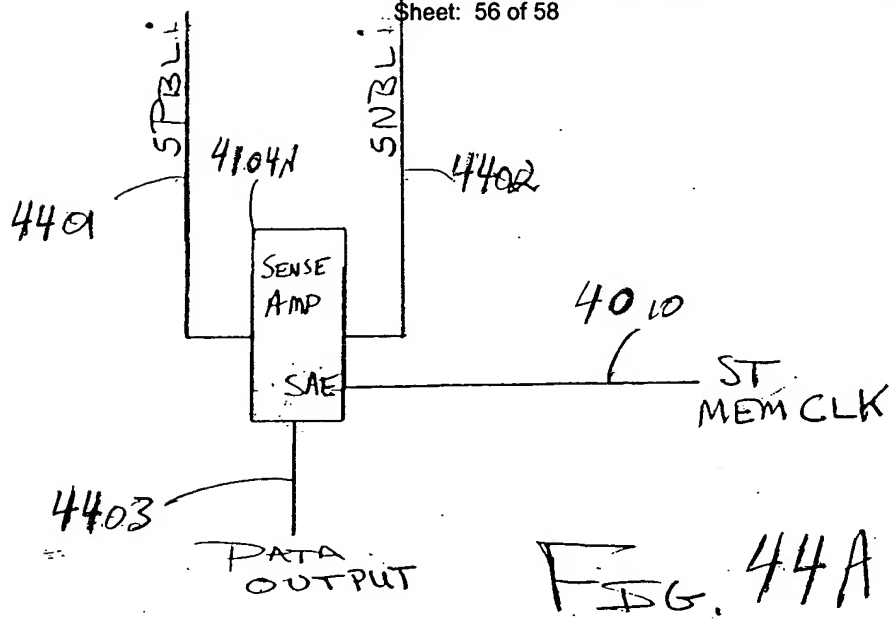


FIG. 44A

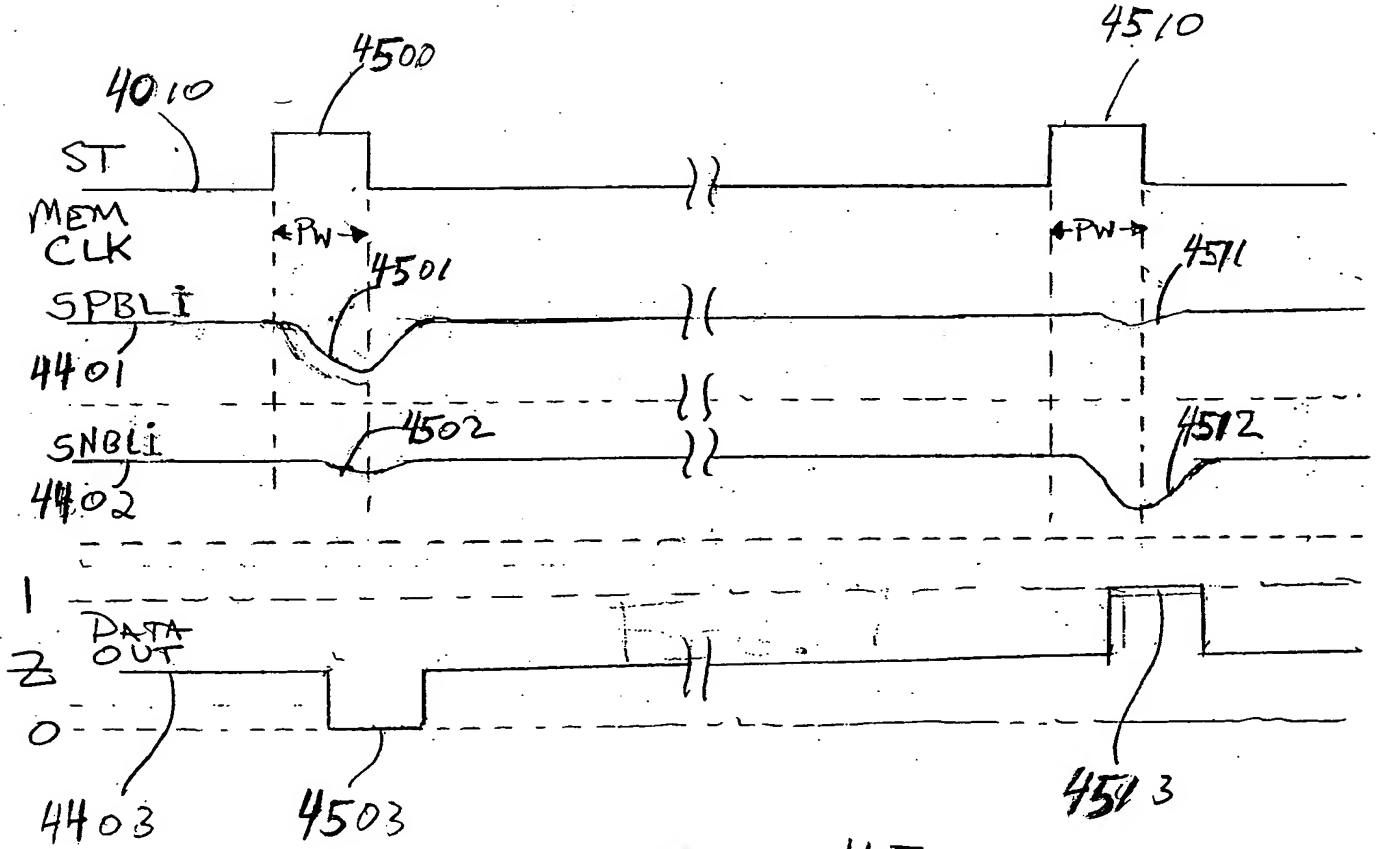


FIG. 45

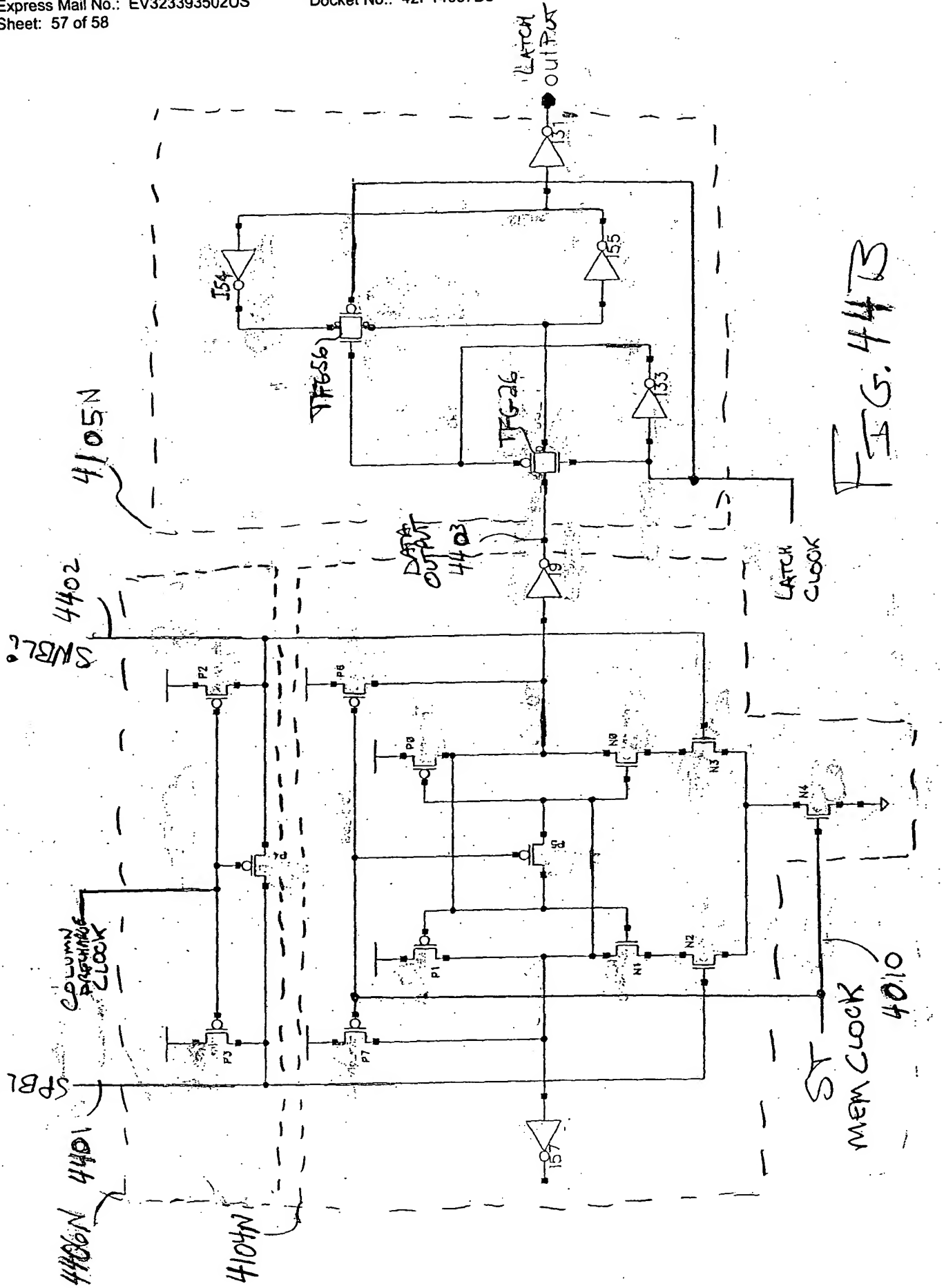

LATCH OUTPUT
I57
I55
I54
TFG56
TFG-26
I53
4105N
DATA OUTPUT 4403
I59
LATCH CLOCK
4402
SBL0
P2
P6
P0
N0
N3
N4
P5
P4
P1
N1
N2
COLUMN DECODE CLOCK
P3
P7
SBL1
4401
4106N
4104N
I57
ST
MEM CLOCK
4010

FIG. 44B

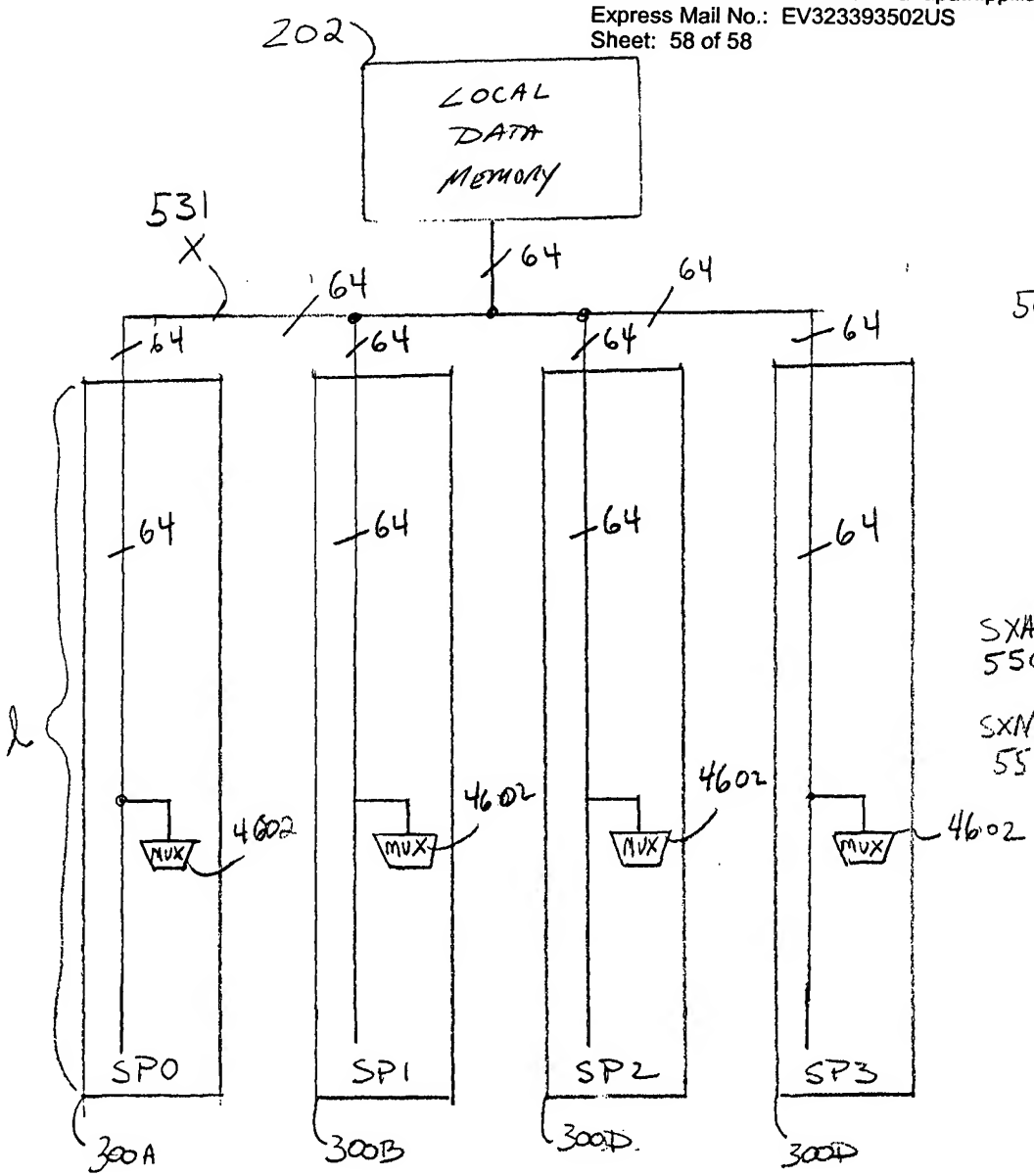


FIG. 46A

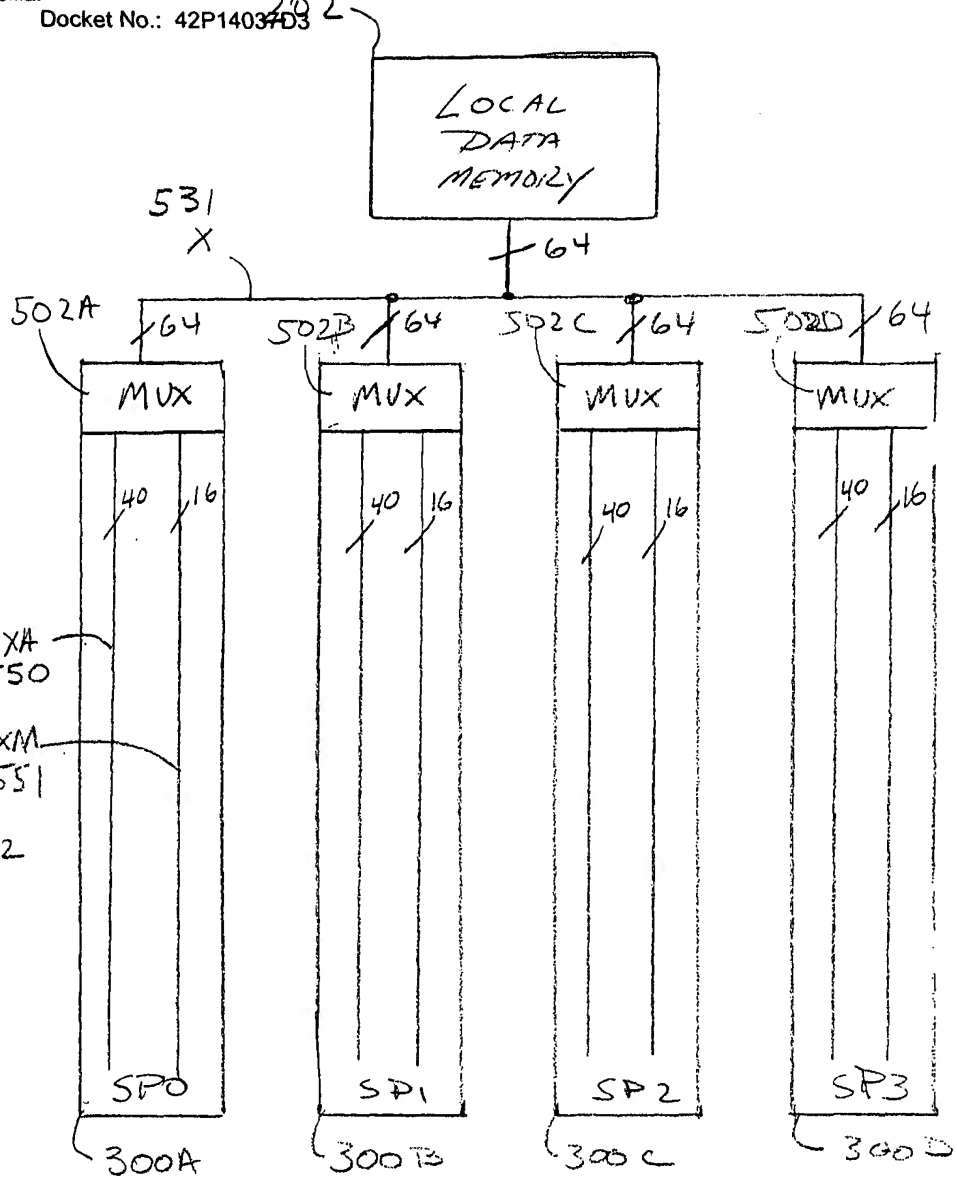


FIG. 46B